Putting Labour Values to Work

Accompanying Simulations

Jonathan F. Cogliano, Roberto Veneziani, and Naoki Yoshihara

Version: January 18, 2022

```
Quiet[SetDirectory[""]]
```

International Exploitation

1970

1970 - Model Setup

Capital Stock Data for Initial Wealth Endowments

In[∘]:=
```
Data = Import["./1970.csv"];
```

In[∘]:=
```
DataNoHeader = SortBy[Drop[Data, 1], #[[5]] &]
```

Out[∘]=
```
{{Rwanda, 684.833, 1.0972, 3.7574, 182.262}, {Myanmar, 7095.89, 1.162, 27.2691, 260.217},
 {Mali, 1655.69, 1.0267, 5.949, 278.314}, {Egypt, 12 224., 1.1736, 34.5139, 354.177},
 {Sierra Leone, 1005.13, 1.0809, 2.7449, 366.181}, {Nepal, 6054.72, 1.0337, 12.0746, 501.442},
 {Ethiopia, 14 377.2, 1.0162, 28.4151, 505.969}, {Burkina Faso, 3240.48, 1.0074, 5.6246, 576.125},
```

```
{Mozambique, 5855.74, 1.125, 9.0227, 649.001}, {Malawi, 3201.31, 1.327, 4.7038, 680.579},
{El Salvador, 2914.64, 1.3158, 3.6731, 793.51}, {Vietnam, 34543.2, 1.4942, 43.4048, 795.839},
{Maldives, 110.866, 1.7161, 0.1157, 958.222}, {Eswatini, 465.919, 1.3378, 0.4313, 1080.27},
{Bangladesh, 69561.5, 1.1678, 64.2325, 1082.96}, {Uganda, 13077., 1.1509, 9.4056, 1390.34},
{China, 1.18378 × 10^6, 1.4501, 827.601, 1430.37}, {Lesotho, 1602.05, 1.5306, 1.0289, 1557.05},
{Burundi, 5587.79, 1.0955, 3.4791, 1606.1}, {Ivory Coast, 8578.77, 1.0401, 5.1021, 1681.42},
{Laos, 4901.46, 1.2059, 2.6884, 1823.19}, {Madagascar, 12725.1, 1.2221, 6.5763, 1935.},
{Morocco, 32474.2, 1.0788, 16.0047, 2029.04}, {Benin, 6074.75, 1.0877, 2.9123, 2085.89},
{Sudan, 22263.3, 1.0822, 10.2817, 2165.33}, {Botswana, 1417.23, 1.1969, 0.6277, 2257.82},
{Pakistan, 135526., 1.1918, 58.1421, 2330.94}, {Zimbabwe, 13707.3, 1.3598, 5.2893, 2591.51},
{Indonesia, 340645., 1.3549, 115.365, 2952.76}, {Tanzania, 40235.3, 1.3049, 13.5355, 2972.58},
{Bulgaria, 26409.5, 2.4006, 8.5076, 3104.22}, {Cameroon, 20824.8, 1.2073, 6.5198, 3194.09},
{Gambia, 1542.92, 1.0591, 0.4644, 3322.4}, {Congo - Brazzaville, 4956.36, 1.166, 1.3269, 3735.29},
{Togo, 7965.76, 1.067, 2.1155, 3765.43}, {Congo - Kinshasa, 76757.2, 1.1156, 20.011, 3835.75},
{India, 2.14358 × 10^6, 1.1807, 555.19, 3860.98}, {Paraguay, 9748.87, 1.6515, 2.4748, 3939.26},
{Kenya, 45093., 1.2869, 11.3014, 3990.03}, {Sri Lanka, 50783.8, 1.9544, 12.4857, 4067.35},
{Jordan, 7272.82, 1.3516, 1.7213, 4225.19}, {Honduras, 11635.7, 1.467, 2.7167, 4283.02},
{Central African Republic, 7866.52, 1.072, 1.8111, 4343.5}, {Tunisia, 22049.4, 1.1441, 5.0638, 4354.31},
{Mongolia, 5579.42, 1.4506, 1.2788, 4363.02}, {Haiti, 20457.7, 1.1606, 4.6762, 4374.86},
{Iraq, 44804.8, 1.0915, 9.918, 4517.53}, {Belize, 553.078, 2.3662, 0.1222, 4526.01},
{Liberia, 6590.44, 1.1421, 1.4007, 4705.11}, {Guatemala, 26660.1, 1.1987, 5.6218, 4742.27},
{Peru, 63862.8, 1.6354, 13.4598, 4744.7}, {Cambodia, 33972.1, 1.1805, 6.9966, 4855.52},
{Zambia, 21232.5, 1.4627, 4.1791, 5080.64}, {Romania, 104467., 2.1014, 20.5489, 5083.84},
{Thailand, 192298., 1.4336, 36.8845, 5213.52}, {Mauritius, 4391.9, 1.5016, 0.8264, 5314.5},
{Bolivia, 23934.1, 1.649, 4.484, 5337.68}, {Fiji, 2795.82, 1.7122, 0.5206, 5370.38},
{Senegal, 27892.8, 1.0538, 4.2575, 6551.46}, {Philippines, 249763., 1.7141, 35.8036, 6975.93},
{Argentina, 175525., 2.0752, 23.8806, 7350.11}, {Brazil, 724635., 1.5132, 95.1133, 7618.65},
{Dominican Republic, 36440.3, 1.5036, 4.4997, 8098.38}, {Panama, 12684.3, 1.9181, 1.5193, 8348.79},
{Syria, 53382., 1.3272, 6.3505, 8405.96}, {Niger, 40577.1, 1.0095, 4.5106, 8995.94},
{Taiwan, 138159., 1.6559, 14.5829, 9474.03}, {South Korea, 313159., 1.9766, 32.1957, 9726.73},
{Namibia, 8391.25, 1.579, 0.8175, 10264.5}, {Costa Rica, 19656.7, 1.6245, 1.8474, 10640.2},
```

```
{Albania, 24195.7, 1.5588, 2.1507, 11250.1}, {Malta, 3615.93, 1.9414, 0.3205, 11282.1},
{Gabon, 6695.38, 1.1438, 0.5893, 11361.6}, {Macao, 2962.01, 1.6243, 0.2462, 12030.9},
{Nicaragua, 30149.6, 1.4265, 2.4065, 12528.4}, {Mauritania, 14560.5, 1.2267, 1.1468, 12696.7},
{Nigeria, 755270., 1.1517, 55.9821, 13491.3}, {Trinidad and Tobago, 13065.1, 2.0285, 0.9454, 13819.7},
{Malaysia, 157570., 1.4985, 10.8041, 14584.2}, {Poland, 478305., 2.3005, 32.6393, 14654.3},
{Ecuador, 92849.8, 1.7755, 6.0694, 15298.}, {Chile, 150597., 2.126, 9.7831, 15393.6},
{Barbados, 3802.22, 2.5157, 0.2389, 15915.5}, {Singapore, 33458.1, 1.6554, 2.0723, 16145.4},
{Uruguay, 46188.9, 1.9377, 2.8098, 16438.5}, {Ghana, 145399., 1.272, 8.7355, 16644.6},
{Hungary, 184918., 2.4364, 10.3661, 17838.7}, {Turkey, 645159., 1.3112, 34.8763, 18498.5},
{South Africa, 420443., 1.7942, 22.0698, 19050.6}, {Angola, 117665., 1.0157, 5.8904, 19975.8},
{Iran, 593994., 1.108, 28.5139, 20831.7}, {Hong Kong, 80453.5, 1.9194, 3.8489, 20903.},
{Jamaica, 40624.2, 1.9935, 1.8756, 21659.3}, {Portugal, 202332., 1.3964, 8.6513, 23387.5},
{Mexico, 1.24804 × 10^6, 1.7327, 51.4936, 24236.8}, {Ireland, 72858.6, 2.3992, 2.9084, 25051.1},
{Venezuela, 289324., 1.3804, 11.3964, 25387.3}, {Colombia, 547307., 1.6843, 21.4801, 25479.7},
{Algeria, 491990., 1.1713, 14.465, 34012.4}, {Bahrain, 7387.42, 1.2862, 0.2126, 34748.},
{Spain, 1.20277 × 10^6, 2.0537, 33.8837, 35497.1}, {Austria, 289522., 2.6166, 7.5162, 38519.8},
{Kuwait, 28951.6, 1.4753, 0.7445, 38887.3}, {Japan, 4.21755 × 10^6, 2.7995, 104.929, 40194.2},
{Saudi Arabia, 265771., 1.5327, 5.8364, 45536.8}, {New Zealand, 128958., 3.0744, 2.8184, 45755.7},
{Greece, 440469., 2.0871, 8.6636, 50841.3}, {Israel, 145633., 2.6072, 2.8137, 51758.6},
{Italy, 2865404, 2.0641, 53.519, 53539.9}, {Cyprus, 38208.2, 1.9709, 0.6136, 62268.9},
{United Kingdom, 3.49714 × 10^6, 2.7415, 55.5735, 62928.1}, {Canada, 1.4935 × 10^6, 2.8333, 21.3743, 69873.6},
{Qatar, 7679.15, 1.5476, 0.1095, 70129.2}, {Finland, 337144., 2.4107, 4.6124, 73095.1},
{Belgium, 726826., 2.3585, 9.6322, 75457.9}, {Denmark, 382183., 2.7572, 4.9312, 77503.},
{Norway, 302917., 2.7742, 3.8761, 78149.9}, {France, 4080021, 2.3108, 51.9577, 78525.8},
{Sweden, 637021., 2.7225, 8.0549, 79084.9}, {Germany, 6464985, 2.9143, 78.5784, 82274.3},
{Australia, 1116206, 2.9684, 12.793, 87251.3}, {Netherlands, 1179288, 2.6553, 13.0019, 90701.2},
{Iceland, 18818.9, 2.2026, 0.2044, 92069.2}, {United States, 21404806, 3.0566, 209.513, 102164.},
{Luxembourg, 42253.9, 2.0601, 0.3397, 124386.}, {Switzerland, 866890., 3.1784, 6.1509, 140937.},
{Brunei, 38352., 1.5899, 0.1295, 296155.}, {United Arab Emirates, 292894., 1.3767, 0.2345, 1.24901 × 10^6}}
```

In[ ]:=
```
CountryList = DataNoHeader[[All, 1]];
```

In[∘]:=
```
InitialCapital = DataNoHeader[[All, 2]]
```

Out[∘]=
```
{684.833, 7095.89, 1655.69, 12 224., 1005.13, 6054.72, 14 377.2, 3240.48, 5855.74, 3201.31, 2914.64, 34 543.2, 110.866,
 465.919, 69 561.5, 13 077., 1.18378 × 10^6, 1602.05, 5587.79, 8578.77, 4901.46, 12 725.1, 32 474.2, 6074.75,
 22 263.3, 1417.23, 135 526., 13 707.3, 340 645., 40 235.3, 26 409.5, 20 824.8, 1542.92, 4956.36, 7965.76, 76 757.2,
 2.14358 × 10^6, 9748.87, 45 093., 50 783.8, 7272.82, 11 635.7, 7866.52, 22 049.4, 5579.42, 20 457.7, 44 804.8, 553.078,
 6590.44, 26 660.1, 63 862.8, 33 972.1, 21 232.5, 104 467., 192 298., 4391.9, 23 934.1, 2795.82, 27 892.8, 249 763.,
 175 525., 724 635., 36 440.3, 12 684.3, 53 382., 40 577.1, 138 159., 313 159., 8391.25, 19 656.7, 24 195.7, 3615.93,
 6695.38, 2962.01, 30 149.6, 14 560.5, 755 270., 13 065.1, 157 570., 478 305., 92 849.8, 150 597., 3802.22, 33 458.1,
 46 188.9, 145 399., 184 918., 645 159., 420 443., 117 665., 593 994., 80 453.5, 40 624.2, 202 332., 1.24804 × 10^6,
 72 858.6, 289 324., 547 307., 491 990., 7387.42, 1.20277 × 10^6, 289 522., 28 951.6, 4.21755 × 10^6, 265 771., 128 958.,
 440 469., 145 633., 2 865 404, 38 208.2, 3.49714 × 10^6, 1.4935 × 10^6, 7679.15, 337 144., 726 826., 382 183., 302 917.,
 4 080 021, 637 021., 6 464 985, 1 116 206, 1 179 288, 18 818.9, 21 404 806, 42 253.9, 866 890., 38 352., 292 894.}
```

In[∘]:=
```
LaborEndowment = DataNoHeader[[All, 3]] × DataNoHeader[[All, 4]] 100 000
```

Out[∘]=
```
{412 262., 3.16867 × 10^6, 610 784., 4.05055 × 10^6, 296 696., 1.24815 × 10^6, 2.88754 × 10^6, 566 622., 1.01505 × 10^6,
 624 194., 483 306., 6.48555 × 10^6, 19 855.3, 57 699.3, 7.50107 × 10^6, 1.08249 × 10^6, 1.2001 × 10^8, 157 483.,
 381 135., 530 669., 324 194., 803 690., 1.72659 × 10^6, 316 771., 1.11269 × 10^6, 75 129.4, 6.92938 × 10^6, 719 239.,
 1.56308 × 10^7, 1.76625 × 10^6, 2.04233 × 10^6, 787 135., 49 184.6, 154 717., 225 724., 2.23243 × 10^6, 6.55513 × 10^7,
 408 713., 1.45438 × 10^6, 2.44021 × 10^6, 232 651., 398 540., 194 150., 579 349., 185 503., 542 720., 1.08255 × 10^6,
 28 915., 159 974., 673 885., 2.20122 × 10^6, 825 949., 611 277., 4.31815 × 10^6, 5.28776 × 10^6, 124 092.,
 739 412., 89 137.1, 448 655., 6.1371 × 10^6, 4.9557 × 10^6, 1.43925 × 10^7, 676 575., 291 417., 842 838., 455 345.,
 2.41478 × 10^6, 6.3638 × 10^6, 129 083., 300 110., 335 251., 62 221.9, 67 404.1, 39 990.3, 343 287., 140 678.,
 6.44746 × 10^6, 191 774., 1.61899 × 10^6, 7.50867 × 10^6, 1.07762 × 10^6, 2.07989 × 10^6, 60 100.1, 343 049., 544 455.,
 1.11116 × 10^6, 2.5256 × 10^6, 4.57298 × 10^6, 3.95976 × 10^6, 598 288., 3.15934 × 10^6, 738 758., 373 901., 1.20807 × 10^6,
 8.9223 × 10^6, 697 783., 1.57316 × 10^6, 3.61789 × 10^6, 1.69429 × 10^6, 27 344.6, 6.9587 × 10^6, 1.96669 × 10^6,
 109 836., 2.9375 × 10^7, 894 545., 866 489., 1.80818 × 10^6, 733 588., 1.10469 × 10^7, 120 934., 1.52355 × 10^7,
 6.05598 × 10^6, 16 946.2, 1.11191 × 10^6, 2.27175 × 10^6, 1.35963 × 10^6, 1.07531 × 10^6, 1.20064 × 10^7, 2.19295 × 10^6,
 2.29001 × 10^7, 3.79747 × 10^6, 3.45239 × 10^6, 45 021.1, 6.40398 × 10^7, 69 981.6, 1.955 × 10^6, 20 589.2, 32 283.6}
```

In[∘]:=
```
Length[InitialCapital]
```
Out[∘]= 128

Initial Parameters

In[∘]:=
```
Clear[N, A, L, b, l, w, p];
N = Length[InitialCapital]; (* Number of agents *)
A = 0.75; (* Technology *)
L = 0.5; (* Labour requirement of technology *)
b = 0.44; (* Subsistence bundle *)
l = LaborEndowment; (* Individual labour endowment *)
```

In[∘]:=
```
T = 1;
```

Countries/Agents and Endowments

In[∘]:=
```
Total[InitialCapital]
```
Out[∘]= 6.61873×10^{7}

In[∘]:=
```
Total[l] ≥ L A^-1 Total[InitialCapital]
```
Out[∘]= True

In[∘]:=
$$\frac{L\,A^{-1}\,\text{Total}[\text{InitialCapital}]}{\text{Total}[l]}$$
Out[∘]= 0.0789092

In[∘]:=
```
Clear[AgentSet];
(* AgentSet=Table[Table[{0,0},{N}],{T}]; *)
AgentSet = Table[{InitialCapital[[i]], LaborEndowment[[i]]}, {i, 1, Length[CountryList]}];
AgentSet // TableForm
```
Out[∘]//TableForm=

684.833	412 262.
7095.89	3.16867×10^{6}

1655.69	610 784.
12 224.	4.05055×10^{6}
1005.13	296 696.
6054.72	1.24815×10^{6}
14 377.2	2.88754×10^{6}
3240.48	566 622.
5855.74	1.01505×10^{6}
3201.31	624 194.
2914.64	483 306.
34 543.2	6.48555×10^{6}
110.866	19 855.3
465.919	57 699.3
69 561.5	7.50107×10^{6}
13 077.	1.08249×10^{6}
1.18378×10^{6}	1.2001×10^{8}
1602.05	157 483.
5587.79	381 135.
8578.77	530 669.
4901.46	324 194.
12 725.1	803 690.
32 474.2	1.72659×10^{6}
6074.75	316 771.
22 263.3	1.11269×10^{6}
1417.23	75 129.4
135 526.	6.92938×10^{6}
13 707.3	719 239.
340 645.	1.56308×10^{7}
40 235.3	1.76625×10^{6}
26 409.5	2.04233×10^{6}
20 824.8	787 135.
1542.92	49 184.6
4956.36	154 717.
7965.76	225 724.
76 757.2	2.23243×10^{6}

2.14358×10^{6}	6.55513×10^{7}
9748.87	408 713.
45 093.	1.45438×10^{6}
50 783.8	2.44021×10^{6}
7272.82	232 651.
11 635.7	398 540.
7866.52	194 150.
22 049.4	579 349.
5579.42	185 503.
20 457.7	542 720.
44 804.8	1.08255×10^{6}
553.078	28 915.
6590.44	159 974.
26 660.1	673 885.
63 862.8	2.20122×10^{6}
33 972.1	825 949.
21 232.5	611 277.
104 467.	4.31815×10^{6}
192 298.	5.28776×10^{6}
4391.9	124 092.
23 934.1	739 412.
2795.82	89 137.1
27 892.8	448 655.
249 763.	6.1371×10^{6}
175 525.	4.9557×10^{6}
724 635.	1.43925×10^{7}
36 440.3	676 575.
12 684.3	291 417.
53 382.	842 838.
40 577.1	455 345.
138 159.	2.41478×10^{6}
313 159.	6.3638×10^{6}
8391.25	129 083.
19 656.7	300 110.

24 195.7	335 251.
3615.93	62 221.9
6695.38	67 404.1
2962.01	39 990.3
30 149.6	343 287.
14 560.5	140 678.
755 270.	6.44746×10^6
13 065.1	191 774.
157 570.	1.61899×10^6
478 305.	7.50867×10^6
92 849.8	1.07762×10^6
150 597.	2.07989×10^6
3802.22	60 100.1
33 458.1	343 049.
46 188.9	544 455.
145 399.	1.11116×10^6
184 918.	2.5256×10^6
645 159.	4.57298×10^6
420 443.	3.95976×10^6
117 665.	598 288.
593 994.	3.15934×10^6
80 453.5	738 758.
40 624.2	373 901.
202 332.	1.20807×10^6
1.24804×10^6	8.9223×10^6
72 858.6	697 783.
289 324.	1.57316×10^6
547 307.	3.61789×10^6
491 990.	1.69429×10^6
7387.42	27 344.6
1.20277×10^6	6.9587×10^6
289 522.	1.96669×10^6
28 951.6	109 836.

4.21755×10^6	2.9375×10^7
265 771.	894 545.
128 958.	866 489.
440 469.	1.80818×10^6
145 633.	733 588.
2 865 404	1.10469×10^7
38 208.2	120 934.
3.49714×10^6	1.52355×10^7
1.4935×10^6	6.05598×10^6
7679.15	16 946.2
337 144.	1.11191×10^6
726 826.	2.27175×10^6
382 183.	1.35963×10^6
302 917.	1.07531×10^6
4 080 021	1.20064×10^7
637 021.	2.19295×10^6
6 464 985	2.29001×10^7
1 116 206	3.79747×10^6
1 179 288	3.45239×10^6
18 818.9	45 021.1
21 404 806	6.40398×10^7
42 253.9	69 981.6
866 890.	1.955×10^6
38 352.	20 589.2
292 894.	32 283.6

Optimisation Routine

Capital Constrained RS

When the simulation is capital constrained and $\mathrm{N}\, l > L\, A^{-1} \sum \omega_{t-1}^{v}$ it is not possible for $z_t^v = l^v$ for all v and $\mathrm{N}\, l > L\, A^{-1} \sum \omega_{t-1}^{v}$ implies that $w_t = b$ and agents are indifferent concerning their labour supply. By Lemma 1 we can analyze solutions to the agents' maximisation problem with $x_t^v = 0$.

Moreover, $w_t = b$ implies $r_t > 0$ and so at any optimal solution it must be $\delta_t^v = 0$ for all v. Thus in the capital constrained case the following procedure can be used to find $(x_t^v; y_t^v; z_t^v)$ given (p_t, r_t):

Step 1: set $x_t^v = \delta_t^v = 0 \; \forall \; v \in \mathcal{N}$

Step 2: set $z_t^v = \frac{L A^{-1} \sum \omega_{t-1}^v}{l_t} l^v \; \forall \; v \in \mathcal{N}$

Step 3: set $y_t^v = A^{-1} \omega_{t-1}^v \; \forall \; v \in \mathcal{N}$

In[∘]:=
```
CapitalConstrained[p_, A_, L_, l_, t_, r_, b_] := (
  Activities⟦t, All, 1⟧ = 0.;
  Activities⟦t, All, 4⟧ = 0.;
  Activities⟦t, All, 2⟧ = (A^-1 (If[t == 1, Total[AgentSet⟦All, 1⟧], Total[Activities⟦t - 1, All, 5⟧]])) l/Total[l];
  Activities⟦t, All, 3⟧ = (If[t == 1, AgentSet⟦All, 1⟧, Activities⟦t - 1, All, 5⟧]);
  Activities⟦t, All, 6⟧ = L Activities⟦t, All, 1⟧ + L Activities⟦t, All, 2⟧;
  Activities⟦t, All, 5⟧ = (1 + r) (A Activities⟦t, All, 1⟧ + Activities⟦t, All, 3⟧) +
    ((1 - A (1 + r))/L - b) L (Activities⟦t, All, 1⟧ + Activities⟦t, All, 2⟧)
  (* 1/p (p Activities⟦t,All,1⟧-p Activities⟦t,All,1⟧+
      (p-(1+r) p A) Activities⟦t,All,2⟧+r Activities⟦t,All,3⟧-p b Activities⟦t,All,6⟧); *)
 )
```

Labour Constrained RS

When the simulation is labour constrained and $\mathrm{N}\, l < L A^{-1} \sum \omega_{t-1}^v$ it is not possible that $A y_t^v = \omega_{t-1}^v$ for all v and $\mathrm{N}\, l < L A^{-1} \sum \omega_{t-1}^v$ implies that $r_t = 0$. Thus agents are indifferent between investing their wealth in productive activities or carrying it to the end of the period and selling it. Hence, $\delta_t^v \in [0, A^{-1} \omega_{t-1}^v]$ is potentially part of the optimal solutions to MP_t^v. By Lemma 1 as in the capital constrained case above, the following procedure can be used to find the optimal $(x_t^v; y_t^v; z_t^v)$ for all v.

Step 1: set $x_t^v = 0 \; \forall \; v \in \mathcal{N}$

Step 2: set $z_t^v = l^v \; \forall \; v \in \mathcal{N}$

Step 3: set $y_t^v = \frac{l_t}{L A^{-1} \sum \omega_{t-1}^v} A^{-1} \omega_{t-1}^v \; \forall \; v \in \mathcal{N}$

Step 4: set $\delta_t^v = \omega_{t-1}^v - \frac{l_t}{L A^{-1} \omega_{t-1}} \omega_{t-1}^v = \left(1 - \frac{l_t}{L A^{-1} \omega_{t-1}}\right) \omega_{t-1}^v \; \forall \in \mathcal{N}$

```
In[•]:= LabourConstrained[p_, A_, L_, l_, t_, r_, b_] := (
  Activities[[t, All, 1]] = 0.;
  Activities[[t, All, 2]] = L^-1 l;
  Activities[[t, All, 3]] = If[t == 1, AgentSet[[All, 1]], Activities[[t - 1, All, 5]]]
      Total[l] / (L A^-1 If[t == 1, Total[AgentSet[[All, 1]]], Total[Activities[[t - 1, All, 5]]]]);
  Activities[[t, All, 4]] = If[t == 1, Total[AgentSet[[All, 1]]], Total[Activities[[t - 1, All, 5]]]] - Activities[[t, All, 3]];
  Activities[[t, All, 6]] = L Activities[[t, All, 1]] + L Activities[[t, All, 2]];
  Activities[[t, All, 5]] = (1 + r) (A Activities[[t, All, 1]] + Activities[[t, All, 3]]) +
    ((1 - A (1 + r)) / L - b) L (Activities[[t, All, 1]] + Activities[[t, All, 2]])
  (* 1/p (p Activities[[t,All,1]]-p Activities[[t,All,1]]+
      (p-(1+r) p A) Activities[[t,All,2]]+r Activities[[t,All,3]]-p b Activities[[t,All,6]]); *)
 )
```

Knife-Edge RS

In the knife-edge case when $N l = L A^{-1} \sum \omega_{t-1}^v$, by Lemma 1 as above, the following steps can be used to calculate the optimal $(x_t^v; y_t^v; z_t^v)$.

Step 1: set $x_t^v = \delta_t^v = 0 \; \forall \; v \in \mathcal{N}$

Step 2: set $z_t^v = l^v \; \forall \; v \in \mathcal{N}$

Step 3: set $y_t^v = A^{-1} \omega_{t-1}^v \; \forall \; v \in \mathcal{N}$

```
In[•]:= KnifeEdge[p_, A_, L_, l_, t_, r_, b_] := (
  Activities⟦t, All, 1⟧ = 0.;
  Activities⟦t, All, 4⟧ = 0.;
  Activities⟦t, All, 2⟧ = L^-1 l;
  Activities⟦t, All, 3⟧ = (If[t == 1, AgentSet⟦All, 1⟧, Activities⟦t - 1, All, 5⟧]);
  Activities⟦t, All, 6⟧ = L Activities⟦t, All, 1⟧ + L Activities⟦t, All, 2⟧;
  Activities⟦t, All, 5⟧ = (1 + r) (A Activities⟦t, All, 1⟧ + Activities⟦t, All, 3⟧) +
    ((1 - A (1 + r))/L - b) L (Activities⟦t, All, 1⟧ + Activities⟦t, All, 2⟧)
  (* 1/p (p Activities⟦t,All,1⟧-p Activities⟦t,All,1⟧+
      (p-(1+r) p A) Activities⟦t,All,2⟧+r Activities⟦t,All,3⟧-p b Activities⟦t,All,6⟧); *)
 )
```

Prices, Wages, and Values

Prices and Wages

A reproducible solution to the simulation procedure consists of a vector (***p***, ***w***) that meets certain conditions described in the associated paper. The determination of $p_t \in \boldsymbol{p}$ and $w_t \in \boldsymbol{w}\ \forall\ t$ has flexibility. Since we are dealing with a single good economy a straightforward choice is $p_t = 1$ for all $t \in T$.

```
In[•]:= (Price = {
  p == (1 + r) p a + w ℓ
    } /. p → 1) // TableForm
```

Out[•]//TableForm=

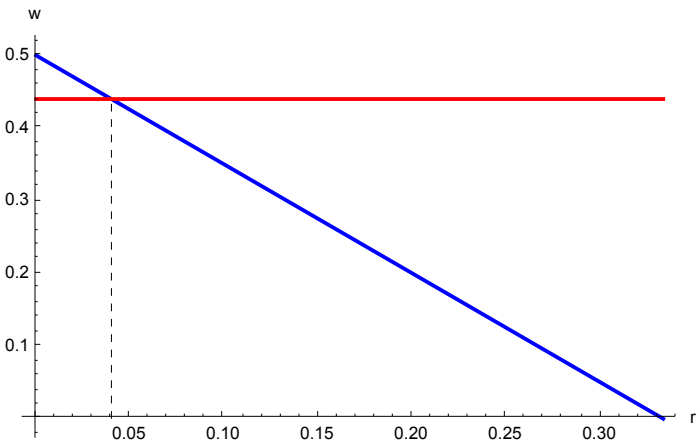
$$1 == a\ (1 + r) + w\ \ell$$

```
In[•]:= TestWage = Solve[Price, w]⟦1⟧ // Simplify
```

Out[•]= $\left\{w \to \frac{1 - a\ (1 + r)}{\ell}\right\}$

In[∘]:=
```
TestWage /. {p → 1, a → A, ℓ → L, r → 0.05}
```
Out[∘]=
```
{w → 0.425}
```

In[∘]:=
```
Quiet[Plot[{w /. Solve[Price, w][[1]] /. {p → 1, a → A, ℓ → L}, b}, {r, 0, (1 - A)/A}, PlotStyle → {{Thick, Blue}, {Thick, Red}},
  AxesLabel → {"r", "w"}, Epilog → {Dashed, Black, Line[{{r /. Solve[Price, r][[1]] /. {p → 1, a → A, ℓ → L} /. w → b, 0},
      {r /. Solve[Price, r][[1]] /. {p → 1, a → A, ℓ → L} /. w → b, b}}]}]]
```
Out[∘]=



In[∘]:=
```
TestProfit = Solve[Price, r][[1]]
```
Out[∘]=
$$\left\{r \to \frac{1 - a - w\,\ell}{a}\right\}$$

Values

In[∘]:=
```
(EmbodiedValues = Flatten[{
      Solve[v == L (1 - A)^-1, v][[1]]
    }]) // TableForm
```
Out[∘]//TableForm=
```
v → 2.
```

Testing that 1 > *v b*

In[∘]:=
```
1 > (v /. EmbodiedValues) b
```

Out[∘]=
```
True
```

1970 - Basic Model

In[∘]:=
```
Clear[Activities, i, Wage, Profit, u];
(Activities = Table[Table[{0.0, 0.0, 0.0, 0.0, 0.0, 0.0}, {N}], {T}]);
Wage = Table[0.0, {T}];
Profit = Table[0.0, {T}];

(* First Stage t=1 simulation *)
(* Setting w1 and π1 *)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]], Wage[[1]] = b, Wage[[1]] = RandomReal[{b, (1 - A (1 + 0))/L}]],
  Wage[[1]] = (1 - A (1 + 0))/L];
Profit[[1]] = ((p - p0 A - Wage[[1]] L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^v,y_t^v,z_t^v,δ_t^v) for all v and t=1*)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]],
   CapitalConstrained[1, A, L, l, 1, Profit[[1]], b], KnifeEdge[1, A, L, l, 1, Profit[[1]], b]],
  LabourConstrained[1, A, L, l, 1, Profit[[1]], b]
 ];
```

```
(* Simulation for remaining T-1 time steps *)
For[t = 2, t ≤ T, t++,

 (* Updating wt *)
 If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧], If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
   Wage⟦t⟧ = b, Wage⟦t⟧ = RandomReal[{b, ((1 - A (1 + r))/L /. r → 0)}]], Wage⟦t⟧ = ((1 - A (1 + r))/L /. r → 0)];

 (* Updating rt *)
 Profit⟦t⟧ = ((p - p0 A - Wage⟦t⟧ L)/(p0 A)) /. {p → 1, p0 → 1};

 (* Checking whether the simulation is capital constrained, labour constrained,
 or on the knife-edge and calling the corresponding function to find optimal (x_t^ν, y_t^ν, z_t^ν, δ_t^ν) for all ν *)
 If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧],
  If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
   CapitalConstrained[1, A, L, l, t, Profit⟦t⟧, b], KnifeEdge[1, A, L, l, t, Profit⟦t⟧, b]],
  LabourConstrained[1, A, L, l, t, Profit⟦t⟧, b]
 ];

]
```

Exploitation and Class Over Time

The chart below captures the number of agents who are exploiters, exploited, or neither over the course of the simulation according to Definition 2, which defines exploitation in relation to Λ_t^v and $v_t\, c_t^v$, where v_t is just the embodied labour value and $c_t^v = \pi_t\, W_{t-1}^v + (w_t - b_t)\; l^v + b_t\, \Lambda_t^v$. An agent v is exploited during t if and only if $\Lambda_t^v > v_t\, c_t^v$, an agent is an exploiter if and only if $\Lambda_t^v < v_t\, c_t^v$, and an agent is neither exploited nor an exploiter if and only if $\Lambda_t^v = v_t\, c_t^v$.

In[∘]:=
```
Clear[IndivExploitation, ConsumptionBundle];
ConsumptionBundle = Table[0.0, {i, T}, {j, N}];
```

```
IndivExploitation = Table[0.0, {i, T}, {j, N}];

For[i = 1, i ≤ N, i++,
 ConsumptionBundle⟦1, i⟧ = EmbodiedValues⟦1, 2⟧ (Profit⟦1⟧ × AgentSet⟦i, 1⟧ + (Wage⟦1⟧ - b) l⟦i⟧ + b Activities⟦1, i, 6⟧);
 IndivExploitation⟦1, i⟧ = Activities⟦1, i, 6⟧ - ConsumptionBundle⟦1, i⟧;

]

For[t = 2, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ConsumptionBundle⟦t, i⟧ =
   EmbodiedValues⟦1, 2⟧ (Profit⟦t⟧ × Activities⟦t - 1, i, 5⟧ + (Wage⟦t⟧ - b) l⟦i⟧ + b Activities⟦t, i, 6⟧);
  IndivExploitation⟦t, i⟧ = Activities⟦t, i, 6⟧ - ConsumptionBundle⟦t, i⟧;
 ]
]

Clear[Exploiters, Exploited, NonExploit];
Exploiters = Table[0.0, {T}];
Exploited = Table[0.0, {T}];
NonExploit = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  If[IndivExploitation⟦t, i⟧ < 0, Exploiters⟦t⟧ ++, If[IndivExploitation⟦t, i⟧ == 0, NonExploit⟦t⟧ ++, Exploited⟦t⟧ ++]];
 ]
]

ExploitationLegend =
  Grid[{{Row[{Graphics[{Thick, Blue, Line[{{0, 0}, {1, 0}}]}], AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploiters", 20], FontFamily → "Times"]}], Spacer[10],
      Row[{Graphics[{Thick, Red, Dotted, Line[{{0, 0}, {1, 0}}]}], AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Neither Exploiter or Exploited", 20], FontFamily → "Times"]}]},
```

```
    {Row[{Graphics[{Thick, Black, Dashed, Line[{{0, 0}, {1, 0}}]}], AspectRatio → 0.15, ImageSize → Scaled[0.04]],
       Spacer[5], Style[Style["Exploited", 20], FontFamily → "Times"]}], Spacer[10],
    }}, Frame → True, Alignment → Left];
(* ExploitationPlot=Labeled[
  ListLinePlot[{Exploiters,Exploited,NonExploit},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black},{Thick,Dotted,Red}},
   Frame→True,FrameLabel→{"t","Total v in Group"},LabelStyle→20,
   PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],ExploitationLegend] *)
```

In[∘]:=
```
Export["./ExploitationPlot.eps", ExploitationPlot, "EPS"]
```

Out[∘]=
```
./ExploitationPlot.eps
```

The chart below captures the class composition of the simulation over time according to Corollary 1 of Theorem 3 (class is defined using labour endowments l^v in relation to means of production).

In[∘]:=
```
Clear[Class1, Class2, Class3, Class4, j, k];
Class1 = Table[0.0, {T}];
Class2 = Table[0.0, {T}];
Class3 = Table[0.0, {T}];
Class4 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ == Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class2⟦j⟧++, 0];
 ]
]
```

```
Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class3⟦j⟧ ++, 0];
 ]
]

Clear[k];
For[k = 1, k ≤ N, k++,
 If[AgentSet⟦k, 1⟧ == 0 && Profit⟦1⟧ > 0, Class4⟦1⟧ ++, 0];
]

Clear[j, k];
For[j = 2, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[Activities⟦j - 1, k, 5⟧ == 0 && Profit⟦j⟧ > 0, Class4⟦j⟧ ++, 0];
 ]
]


(* ClassLegend=
 Column[{
   Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^1 = {∑v ∈ (+,0,+)\(+,0,0); A_t y_t^v < z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,DotDashed,Green,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^2 = {∑v ∈ (+,0,0); A_t y_t^v = z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Purple,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^3 = {∑v ∈ (+,+,0)\(+,0,0); A_t y_t^v > z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Black,Dotted,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^4 = {∑v ∈ (0,+,0); W_{t-1}^v = 0}",20],FontFamily→"Times"]}]
```

```
  },Frame→True,Alignment→Left];
ClassPlotCorollary1=Labeled[
  ListLinePlot[{Class1,Class2,Class3,Class4},PlotStyle→{{Thick,Blue},{Thick,DotDashed,Green},
     {Thick,Dashed,Purple},{Thick, Dotted,Black},{Thick,DotDashed,Orange},{Thick,Dashed,Green}},
   PlotRange→{{1,T},{-10,N+10}},Frame→True,FrameLabel→{"t","Total ν in Classes"},
   LabelStyle→20,ImageSize→{500,350},AxesOrigin→{1,-10}],ClassLegend] *)
```

In[∘]:=
```
(* Export["./ClassPlotCorollary1.eps",ClassPlotCorollary1,"EPS"] *)
```

The chart below captures the intersection of class and exploitation over the simulation. If an agent is in C^1 according to Corollary 1 of Theorem 3 and is also an exploiter according to Definition 2, then the agent is in the group "$C^1 \wedge$ Exploiter". If an agent is in $C^3 \cup C^4$ and is exploited they are in group "$(C^3 \cup C^4) \wedge$ Exploited".

In[∘]:=
```
Clear[CECP1, CECP2, CECP3, j, k];
CECP1 = Table[0.0, {T}];
CECP2 = Table[0.0, {T}];
CECP3a = Table[0.0, {T}];
CECP3b = Table[0.0, {T}];
CECP3 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ < 0, CECP1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
  If[If[j == 1, AgentSet⟦k, 1⟧, Activities⟦j - 1, k, 5⟧] == 0 && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
 ]
]
```

```
(* CECPLegend=
 Grid[{{Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],Spacer[5],
      Style[Style["∑v ∈ C_t^1 ∧ Exploiter",20],FontFamily→"Times"]}],Spacer[10],
    Row[{Graphics[{Thick,Black,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
      Spacer[5],Style[Style["∑v ∈ (C_t^3 ⋃ C_t^4) ∧ Exploited",20],FontFamily→"Times"]}],
   }},Frame→True,Alignment→Left];
CECPPlotCorollary1=
 Labeled[ListLinePlot[{CECP1,CECP3},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black}},Frame→True,FrameLabel→
    {"t","Total v in Group"},LabelStyle→20,PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],CECPLegend] *)
```

In[∘]:=
```
(* Export["./CECPPlotCorollary1.eps",CECPPlotCorollary1,"EPS"] *)
```

The charts below display the distribution of an index of the intensity of exploitation across the agents over the simulation. The index itself is calculated as $e_t^v = \Lambda_t^v / v_t\, c_t^v$ and the charts that follow explore some possibilities for visualizing the intensity of exploitation over time.

In[∘]:=
```
Clear[ExploitationIndex];
ExploitationIndex = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndex[[t, i]] = Activities[[t, i, 6]] / (EmbodiedValues[[1, 2]] ConsumptionBundle[[t, i]] / EmbodiedValues[[1, 2]]) /. p → 1
 ]
]
```

The chart below uses *Mathematica's* `ArrayPlot` function to display the exploitation index of the N agents over *T*. There is a fixed distribution of uneven exploitation intensity while the simulation is capital constrained, however, this pattern disappears once the simulation is labour constrained, at which point exploitation intensity is the same for all $v \in \mathcal{N}$.

In[∘]:=
```
ExploitationIndex[[1, 1]]
```

Out[∘]=
```
1.13419
```

In[∘]:=
```
ExploitationIndex[[1]]
```

Out[∘]=
```
{1.13419, 1.13344, 1.13283, 1.13243, 1.13195, 1.13005, 1.12988, 1.12893, 1.12886, 1.12969, 1.12852, 1.12943,
 1.1291, 1.12589, 1.12435, 1.12077, 1.1236, 1.1232, 1.11749, 1.11559, 1.11691, 1.11601, 1.11226, 1.1118,
 1.11076, 1.11219, 1.11132, 1.11195, 1.10853, 1.1073, 1.11968, 1.10275, 1.09673, 1.09592, 1.09197, 1.09307,
 1.09511, 1.10597, 1.09717, 1.10976, 1.09686, 1.09939, 1.08568, 1.08863, 1.09831, 1.08907, 1.08465, 1.11186,
 1.08487, 1.08683, 1.09961, 1.08495, 1.09264, 1.10555, 1.09067, 1.09184, 1.09551, 1.09673, 1.06041, 1.08547,
 1.09181, 1.07406, 1.06997, 1.0821, 1.05908, 1.03056, 1.06609, 1.0754, 1.05719, 1.05663, 1.04913, 1.06506,
 1.01967, 1.04702, 1.03195, 1.0153, 1.00124, 1.05366, 1.02179, 1.05867, 1.03375, 1.04887, 1.05917, 1.02158,
 1.03519, 0.987495, 1.04797, 0.977488, 1.01251, 0.92645, 0.934046, 1.00968, 1.00995, 0.952563, 0.978653,
 1.01435, 0.937686, 0.967709, 0.851501, 0.86663, 0.947657, 0.971583, 0.871663, 0.975076, 0.84659, 0.97004,
 0.887339, 0.924842, 0.874912, 0.833119, 0.898705, 0.884936, 0.746596, 0.842174, 0.830314, 0.858385, 0.85793,
 0.816647, 0.851425, 0.857475, 0.848898, 0.815456, 0.767, 0.820437, 0.670182, 0.752133, 0.36121, 0.0992267}
```

In[∘]:=
```
CountryList
```

Out[∘]=
```
{Rwanda, Myanmar, Mali, Egypt, Sierra Leone, Nepal, Ethiopia, Burkina Faso, Mozambique, Malawi, El Salvador,
 Vietnam, Maldives, Eswatini, Bangladesh, Uganda, China, Lesotho, Burundi, Ivory Coast, Laos, Madagascar,
 Morocco, Benin, Sudan, Botswana, Pakistan, Zimbabwe, Indonesia, Tanzania, Bulgaria, Cameroon, Gambia,
 Congo - Brazzaville, Togo, Congo - Kinshasa, India, Paraguay, Kenya, Sri Lanka, Jordan, Honduras,
 Central African Republic, Tunisia, Mongolia, Haiti, Iraq, Belize, Liberia, Guatemala, Peru, Cambodia, Zambia,
 Romania, Thailand, Mauritius, Bolivia, Fiji, Senegal, Philippines, Argentina, Brazil, Dominican Republic,
 Panama, Syria, Niger, Taiwan, South Korea, Namibia, Costa Rica, Albania, Malta, Gabon, Macao, Nicaragua,
 Mauritania, Nigeria, Trinidad and Tobago, Malaysia, Poland, Ecuador, Chile, Barbados, Singapore, Uruguay,
 Ghana, Hungary, Turkey, South Africa, Angola, Iran, Hong Kong, Jamaica, Portugal, Mexico, Ireland, Venezuela,
 Colombia, Algeria, Bahrain, Spain, Austria, Kuwait, Japan, Saudi Arabia, New Zealand, Greece, Israel,
 Italy, Cyprus, United Kingdom, Canada, Qatar, Finland, Belgium, Denmark, Norway, France, Sweden, Germany,
 Australia, Netherlands, Iceland, United States, Luxembourg, Switzerland, Brunei, United Arab Emirates}
```

In[∘]:=
```
(ExploitTable = Table[{CountryList[[i]], ExploitationIndex[[1, i]], DataNoHeader[[i, 5]]}, {i, 1, N}]) // TableForm
Export["./ExploitTable1970.csv", ExploitTable, "CSV"]
```

Out[∘]//TableForm=

Rwanda	1.13419	182.262
Myanmar	1.13344	260.217

Mali	1.13283	278.314
Egypt	1.13243	354.177
Sierra Leone	1.13195	366.181
Nepal	1.13005	501.442
Ethiopia	1.12988	505.969
Burkina Faso	1.12893	576.125
Mozambique	1.12886	649.001
Malawi	1.12969	680.579
El Salvador	1.12852	793.51
Vietnam	1.12943	795.839
Maldives	1.1291	958.222
Eswatini	1.12589	1080.27
Bangladesh	1.12435	1082.96
Uganda	1.12077	1390.34
China	1.1236	1430.37
Lesotho	1.1232	1557.05
Burundi	1.11749	1606.1
Ivory Coast	1.11559	1681.42
Laos	1.11691	1823.19
Madagascar	1.11601	1935.
Morocco	1.11226	2029.04
Benin	1.1118	2085.89
Sudan	1.11076	2165.33
Botswana	1.11219	2257.82
Pakistan	1.11132	2330.94
Zimbabwe	1.11195	2591.51
Indonesia	1.10853	2952.76
Tanzania	1.1073	2972.58
Bulgaria	1.11968	3104.22
Cameroon	1.10275	3194.09
Gambia	1.09673	3322.4
Congo – Brazzaville	1.09592	3735.29
Togo	1.09197	3765.43
Congo – Kinshasa	1.09307	3835.75
India	1.09511	3860.98
Paraguay	1.10597	3939.26

Kenya	1.09717	3990.03
Sri Lanka	1.10976	4067.35
Jordan	1.09686	4225.19
Honduras	1.09939	4283.02
Central African Republic	1.08568	4343.5
Tunisia	1.08863	4354.31
Mongolia	1.09831	4363.02
Haiti	1.08907	4374.86
Iraq	1.08465	4517.53
Belize	1.11186	4526.01
Liberia	1.08487	4705.11
Guatemala	1.08683	4742.27
Peru	1.09961	4744.7
Cambodia	1.08495	4855.52
Zambia	1.09264	5080.64
Romania	1.10555	5083.84
Thailand	1.09067	5213.52
Mauritius	1.09184	5314.5
Bolivia	1.09551	5337.68
Fiji	1.09673	5370.38
Senegal	1.06041	6551.46
Philippines	1.08547	6975.93
Argentina	1.09181	7350.11
Brazil	1.07406	7618.65
Dominican Republic	1.06997	8098.38
Panama	1.0821	8348.79
Syria	1.05908	8405.96
Niger	1.03056	8995.94
Taiwan	1.06609	9474.03
South Korea	1.0754	9726.73
Namibia	1.05719	10 264.5
Costa Rica	1.05663	10 640.2
Albania	1.04913	11 250.1
Malta	1.06506	11 282.1
Gabon	1.01967	11 361.6
Macao	1.04702	12 030.9

Nicaragua	1.03195	12 528.4
Mauritania	1.0153	12 696.7
Nigeria	1.00124	13 491.3
Trinidad and Tobago	1.05366	13 819.7
Malaysia	1.02179	14 584.2
Poland	1.05867	14 654.3
Ecuador	1.03375	15 298.
Chile	1.04887	15 393.6
Barbados	1.05917	15 915.5
Singapore	1.02158	16 145.4
Uruguay	1.03519	16 438.5
Ghana	0.987495	16 644.6
Hungary	1.04797	17 838.7
Turkey	0.977488	18 498.5
South Africa	1.01251	19 050.6
Angola	0.92645	19 975.8
Iran	0.934046	20 831.7
Hong Kong	1.00968	20 903.
Jamaica	1.00995	21 659.3
Portugal	0.952563	23 387.5
Mexico	0.978653	24 236.8
Ireland	1.01435	25 051.1
Venezuela	0.937686	25 387.3
Colombia	0.967709	25 479.7
Algeria	0.851501	34 012.4
Bahrain	0.86663	34 748.
Spain	0.947657	35 497.1
Austria	0.971583	38 519.8
Kuwait	0.871663	38 887.3
Japan	0.975076	40 194.2
Saudi Arabia	0.84659	45 536.8
New Zealand	0.97004	45 755.7
Greece	0.887339	50 841.3
Israel	0.924842	51 758.6
Italy	0.874912	53 539.9
Cyprus	0.833119	62 268.9

United Kingdom	0.898705	62 928.1
Canada	0.884936	69 873.6
Qatar	0.746596	70 129.2
Finland	0.842174	73 095.1
Belgium	0.830314	75 457.9
Denmark	0.858385	77 503.
Norway	0.85793	78 149.9
France	0.816647	78 525.8
Sweden	0.851425	79 084.9
Germany	0.857475	82 274.3
Australia	0.848898	87 251.3
Netherlands	0.815456	90 701.2
Iceland	0.767	92 069.2
United States	0.820437	102 164.
Luxembourg	0.670182	124 386.
Switzerland	0.752133	140 937.
Brunei	0.36121	296 155.
United Arab Emirates	0.0992267	1.24901×10^6

Out[•]= ./ExploitTable1970.csv

In[•]:=
```
(* ExploitationArray=ArrayPlot[Transpose[ExploitationIndex],
  FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./ExploitationArray.eps",ExploitationArray,"EPS"] *)
```

In[•]:=
```
ExploitationIndexDistrib = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndexDistrib[[t, i]] = {t, ExploitationIndex[[t, i]]}
 ]
]
```

Below the Gini coefficient for the exploitation intensity index is plotted over *T*. The pattern of the Gini coefficient is consistent with the previous charts depicting the exploitation intensity index.

In[•]:=
```
(* Gini=Table[0.0,{T}];
For[t=1,t≤T,t++,
 Gini[[t]]=N/(N-1) ((Σ_{i=1}^N Σ_{j=1}^N Abs[ExploitationIndex[[t,i]]-ExploitationIndex[[t,j]]])/(2 N^2 Mean[ExploitationIndex[[t]]]))
] *)
```

In[•]:=
```
Clear[SortExploitationIndex];
SortExploitationIndex = Table[Table[0.0, {N}], {t, 1, T}];
GiniTest = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortExploitationIndex[[t]] = Sort[ExploitationIndex[[t]]];
 GiniTest[[t]] = N/(N - 1) Σ_{i=1}^N ((2 i - N - 1) SortExploitationIndex[[t, i]]) / (N^2 Mean[SortExploitationIndex[[t]]])
]
```

In[•]:=
```
Chop[GiniTest]
```

Out[•]= {0.0664079}

In[•]:=
```
(* ExploitationGini=ListLinePlot[GiniTest,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: e_t^y"}] *)
```

In[•]:=
```
(* Export["./ExploitationGini.eps",ExploitationGini,"EPS"] *)
```

Plotting Gini coefficient for wealth $W_{t-1} = p_{t-1}\,\omega_{t-1}$.

In[∘]:=
```
Clear[SortWealth, WealthGini];
SortWealth = Table[Table[0.0, {N}], {t, 1, T}];
WealthGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortWealth[[t]] = If[t == 1, Sort[AgentSet[[All, 1]]], Sort[Activities[[t - 1, All, 5]]]];
 WealthGini[[t]] = N/(N - 1) Sum[(2 i - N - 1) SortWealth[[t, i]]/(N^2 Mean[SortWealth[[t]]]), {i, 1, N}]
]
(* WealthGiniPlot=
 ListLinePlot[WealthGini,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: W^γ_{t-1}"},LabelStyle→12,PlotRange→{0,1}] *)
```

In[∘]:=
```
(* Export["./WealthGini.eps",WealthGiniPlot,"EPS"] *)
```

In[∘]:=
```
WealthGini
```

Out[∘]=
```
{0.868673}
```

The figure below plots the distribution of wealth for select *t*.

In[∘]:=
```
(* WealthDistribRow=GraphicsRow[{
    Histogram[AgentSet[[All,1]],10,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 1",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[24,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 25",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[49,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 50",18],AxesLabel→{"ω^γ_{t-1}"}]
   },Spacings→{15,-20}
  ]
  Export["./WealthDistribRow.eps",WealthDistribRow,"EPS"]
*)
```

Income Figures

Figures on net and gross income.

In[•]:=
```
Clear[Income];
Income = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  Income⟦t, i⟧ = Profit⟦t⟧ × Activities⟦t, i, 5⟧ + Wage⟦t⟧ × Activities⟦t, i, 6⟧
 ]
]

IncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
```

$$\text{IncomeGini}[[t]] = \frac{N}{N-1}\left(\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\text{Abs}[\text{Income}[[t, i]] - \text{Income}[[t, j]]]\right) \Big/ \left(2\,N^2\,\text{Mean}[\text{Income}[[t]]]\right)\right)$$

```
]
(* IncomeGiniPlot=ListLinePlot[IncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[IncomeGini]+0.1},AxesLabel→{"t","Gini: Net Income"},LabelStyle→14] *)
```

In[•]:=
```
(* Export["./NetIncomeGini.eps",IncomeGiniPlot,"EPS"] *)
```

In[•]:=
```
Clear[IncomeShares];
IncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  IncomeShares[[t, i]] = Income[[t, i]] / Total[Income[[t]]]
 ]
]
(* IncomeArray=ArrayPlot[Transpose[IncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},
  PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",ColorFunctionScaling→True,
  DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./NetIncomeArray.eps",IncomeArray,"EPS"] *)
```

In[•]:=
```
Clear[GrossIncome];
GrossIncome = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncome[[t, i]] = (1 + Profit[[t]]) Activities[[t, i, 5]] + Wage[[t]] × Activities[[t, i, 6]]
 ]
]

GrossIncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 GrossIncomeGini[[t]] = N/(N - 1) ((Sum[Sum[Abs[GrossIncome[[t, i]] - GrossIncome[[t, j]]], {j, 1, N}], {i, 1, N}]) / (2 N^2 Mean[GrossIncome[[t]]]))
]
(* GrossIncomeGiniPlot=ListLinePlot[GrossIncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[GrossIncomeGini]+0.1},AxesLabel→{"t","Gini: Income"},LabelStyle→14] *)
```

```
In[•]:= (* Export["./GrossIncomeGini.eps",GrossIncomeGiniPlot,"EPS"] *)

In[•]:= GrossIncomeGini

Out[•]= {0.841791}

In[•]:= Clear[GrossIncomeShares];
GrossIncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncomeShares[[t, i]] = GrossIncome[[t, i]] / Total[GrossIncome[[t]]]
 ]
]
(* GrossIncomeArray=
 ArrayPlot[Transpose[GrossIncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./GrossIncomeArray.eps",GrossIncomeArray,"EPS"] *)
```

Updated Simulation Reporting

```
In[•]:= (ExploitedTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploitedTable1970.csv", ExploitedTable, "CSV"]

Out[•]//TableForm=
Rwanda          &   1.13419   \\
Myanmar         &   1.13344   \\
Mali            &   1.13283   \\
Egypt           &   1.13243   \\
Sierra Leone    &   1.13195   \\
Nepal           &   1.13005   \\
Ethiopia        &   1.12988   \\
```

```
Burkina Faso              &   1.12893   \\
Mozambique                &   1.12886   \\
Malawi                    &   1.12969   \\
El Salvador               &   1.12852   \\
Vietnam                   &   1.12943   \\
Maldives                  &   1.1291    \\
Eswatini                  &   1.12589   \\
Bangladesh                &   1.12435   \\
Uganda                    &   1.12077   \\
China                     &   1.1236    \\
Lesotho                   &   1.1232    \\
Burundi                   &   1.11749   \\
Ivory Coast               &   1.11559   \\
Laos                      &   1.11691   \\
Madagascar                &   1.11601   \\
Morocco                   &   1.11226   \\
Benin                     &   1.1118    \\
Sudan                     &   1.11076   \\
Botswana                  &   1.11219   \\
Pakistan                  &   1.11132   \\
Zimbabwe                  &   1.11195   \\
Indonesia                 &   1.10853   \\
Tanzania                  &   1.1073    \\
Bulgaria                  &   1.11968   \\
Cameroon                  &   1.10275   \\
Gambia                    &   1.09673   \\
Congo – Brazzaville       &   1.09592   \\
Togo                      &   1.09197   \\
Congo – Kinshasa          &   1.09307   \\
India                     &   1.09511   \\
Paraguay                  &   1.10597   \\
Kenya                     &   1.09717   \\
Sri Lanka                 &   1.10976   \\
Jordan                    &   1.09686   \\
Honduras                  &   1.09939   \\
Central African Republic  &   1.08568   \\
```

```
Tunisia                  &   1.08863   \\
Mongolia                 &   1.09831   \\
Haiti                    &   1.08907   \\
Iraq                     &   1.08465   \\
Belize                   &   1.11186   \\
Liberia                  &   1.08487   \\
Guatemala                &   1.08683   \\
Peru                     &   1.09961   \\
Cambodia                 &   1.08495   \\
Zambia                   &   1.09264   \\
Romania                  &   1.10555   \\
Thailand                 &   1.09067   \\
Mauritius                &   1.09184   \\
Bolivia                  &   1.09551   \\
Fiji                     &   1.09673   \\
Senegal                  &   1.06041   \\
Philippines              &   1.08547   \\
Argentina                &   1.09181   \\
Brazil                   &   1.07406   \\
Dominican Republic       &   1.06997   \\
Panama                   &   1.0821    \\
Syria                    &   1.05908   \\
Niger                    &   1.03056   \\
Taiwan                   &   1.06609   \\
South Korea              &   1.0754    \\
Namibia                  &   1.05719   \\
Costa Rica               &   1.05663   \\
Albania                  &   1.04913   \\
Malta                    &   1.06506   \\
Gabon                    &   1.01967   \\
Macao                    &   1.04702   \\
Nicaragua                &   1.03195   \\
Mauritania               &   1.0153    \\
Nigeria                  &   1.00124   \\
Trinidad and Tobago      &   1.05366   \\
Malaysia                 &   1.02179   \\
```

```
Poland                    &    1.05867    \\
Ecuador                   &    1.03375    \\
Chile                     &    1.04887    \\
Barbados                  &    1.05917    \\
Singapore                 &    1.02158    \\
Uruguay                   &    1.03519    \\
Hungary                   &    1.04797    \\
South Africa              &    1.01251    \\
Hong Kong                 &    1.00968    \\
Jamaica                   &    1.00995    \\
Ireland                   &    1.01435    \\
```

Out[•]= ./ExploitedTable1970.csv

In[•]:=
```
(ExploiterTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploiterTable1970.csv", ExploiterTable, "CSV"]
```

Out[•]//TableForm=
```
Ghana                     &    0.987495     \\
Turkey                    &    0.977488     \\
Angola                    &    0.92645      \\
Iran                      &    0.934046     \\
Portugal                  &    0.952563     \\
Mexico                    &    0.978653     \\
Venezuela                 &    0.937686     \\
Colombia                  &    0.967709     \\
Algeria                   &    0.851501     \\
Bahrain                   &    0.86663      \\
Spain                     &    0.947657     \\
Austria                   &    0.971583     \\
Kuwait                    &    0.871663     \\
Japan                     &    0.975076     \\
Saudi Arabia              &    0.84659      \\
New Zealand               &    0.97004      \\
Greece                    &    0.887339     \\
Israel                    &    0.924842     \\
Italy                     &    0.874912     \\
```

```
Cyprus                  &    0.833119     \\
United Kingdom          &    0.898705     \\
Canada                  &    0.884936     \\
Qatar                   &    0.746596     \\
Finland                 &    0.842174     \\
Belgium                 &    0.830314     \\
Denmark                 &    0.858385     \\
Norway                  &    0.85793      \\
France                  &    0.816647     \\
Sweden                  &    0.851425     \\
Germany                 &    0.857475     \\
Australia               &    0.848898     \\
Netherlands             &    0.815456     \\
Iceland                 &    0.767        \\
United States           &    0.820437     \\
Luxembourg              &    0.670182     \\
Switzerland             &    0.752133     \\
Brunei                  &    0.36121      \\
United Arab Emirates    &    0.0992267    \\
```

Out[•]= ./ExploiterTable1970.csv

In[•]:=
```
(ExploitIncomeCSV = Table[{CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"}, {i, 1, N}]) // TableForm
Export["./ExploitIncomeTable1970.csv", ExploitIncomeCSV, "CSV"]
```

Out[•]//TableForm=
```
Rwanda                      &    1.13419      \\
Myanmar                     &    1.13344      \\
Mali                        &    1.13283      \\
Egypt                       &    1.13243      \\
Sierra Leone                &    1.13195      \\
Nepal                       &    1.13005      \\
Ethiopia                    &    1.12988      \\
Burkina Faso                &    1.12893      \\
Mozambique                  &    1.12886      \\
Malawi                      &    1.12969      \\
El Salvador                 &    1.12852      \\
Vietnam                     &    1.12943      \\
```

```
Maldives                    &   1.1291      \\
Eswatini                    &   1.12589     \\
Bangladesh                  &   1.12435     \\
Uganda                      &   1.12077     \\
China                       &   1.1236      \\
Lesotho                     &   1.1232      \\
Burundi                     &   1.11749     \\
Ivory Coast                 &   1.11559     \\
Laos                        &   1.11691     \\
Madagascar                  &   1.11601     \\
Morocco                     &   1.11226     \\
Benin                       &   1.1118      \\
Sudan                       &   1.11076     \\
Botswana                    &   1.11219     \\
Pakistan                    &   1.11132     \\
Zimbabwe                    &   1.11195     \\
Indonesia                   &   1.10853     \\
Tanzania                    &   1.1073      \\
Bulgaria                    &   1.11968     \\
Cameroon                    &   1.10275     \\
Gambia                      &   1.09673     \\
Congo - Brazzaville         &   1.09592     \\
Togo                        &   1.09197     \\
Congo - Kinshasa            &   1.09307     \\
India                       &   1.09511     \\
Paraguay                    &   1.10597     \\
Kenya                       &   1.09717     \\
Sri Lanka                   &   1.10976     \\
Jordan                      &   1.09686     \\
Honduras                    &   1.09939     \\
Central African Republic    &   1.08568     \\
Tunisia                     &   1.08863     \\
Mongolia                    &   1.09831     \\
Haiti                       &   1.08907     \\
Iraq                        &   1.08465     \\
Belize                      &   1.11186     \\
```

```
Liberia                    &    1.08487      \\
Guatemala                  &    1.08683      \\
Peru                       &    1.09961      \\
Cambodia                   &    1.08495      \\
Zambia                     &    1.09264      \\
Romania                    &    1.10555      \\
Thailand                   &    1.09067      \\
Mauritius                  &    1.09184      \\
Bolivia                    &    1.09551      \\
Fiji                       &    1.09673      \\
Senegal                    &    1.06041      \\
Philippines                &    1.08547      \\
Argentina                  &    1.09181      \\
Brazil                     &    1.07406      \\
Dominican Republic         &    1.06997      \\
Panama                     &    1.0821       \\
Syria                      &    1.05908      \\
Niger                      &    1.03056      \\
Taiwan                     &    1.06609      \\
South Korea                &    1.0754       \\
Namibia                    &    1.05719      \\
Costa Rica                 &    1.05663      \\
Albania                    &    1.04913      \\
Malta                      &    1.06506      \\
Gabon                      &    1.01967      \\
Macao                      &    1.04702      \\
Nicaragua                  &    1.03195      \\
Mauritania                 &    1.0153       \\
Nigeria                    &    1.00124      \\
Trinidad and Tobago        &    1.05366      \\
Malaysia                   &    1.02179      \\
Poland                     &    1.05867      \\
Ecuador                    &    1.03375      \\
Chile                      &    1.04887      \\
Barbados                   &    1.05917      \\
Singapore                  &    1.02158      \\
```

```
Uruguay                 &   1.03519     \\
Ghana                   &   0.987495    \\
Hungary                 &   1.04797     \\
Turkey                  &   0.977488    \\
South Africa            &   1.01251     \\
Angola                  &   0.92645     \\
Iran                    &   0.934046    \\
Hong Kong               &   1.00968     \\
Jamaica                 &   1.00995     \\
Portugal                &   0.952563    \\
Mexico                  &   0.978653    \\
Ireland                 &   1.01435     \\
Venezuela               &   0.937686    \\
Colombia                &   0.967709    \\
Algeria                 &   0.851501    \\
Bahrain                 &   0.86663     \\
Spain                   &   0.947657    \\
Austria                 &   0.971583    \\
Kuwait                  &   0.871663    \\
Japan                   &   0.975076    \\
Saudi Arabia            &   0.84659     \\
New Zealand             &   0.97004     \\
Greece                  &   0.887339    \\
Israel                  &   0.924842    \\
Italy                   &   0.874912    \\
Cyprus                  &   0.833119    \\
United Kingdom          &   0.898705    \\
Canada                  &   0.884936    \\
Qatar                   &   0.746596    \\
Finland                 &   0.842174    \\
Belgium                 &   0.830314    \\
Denmark                 &   0.858385    \\
Norway                  &   0.85793     \\
France                  &   0.816647    \\
Sweden                  &   0.851425    \\
Germany                 &   0.857475    \\
```

```
Australia                  &    0.848898     \\
Netherlands                &    0.815456     \\
Iceland                    &    0.767        \\
United States              &    0.820437     \\
Luxembourg                 &    0.670182     \\
Switzerland                &    0.752133     \\
Brunei                     &    0.36121      \\
United Arab Emirates       &    0.0992267    \\
```

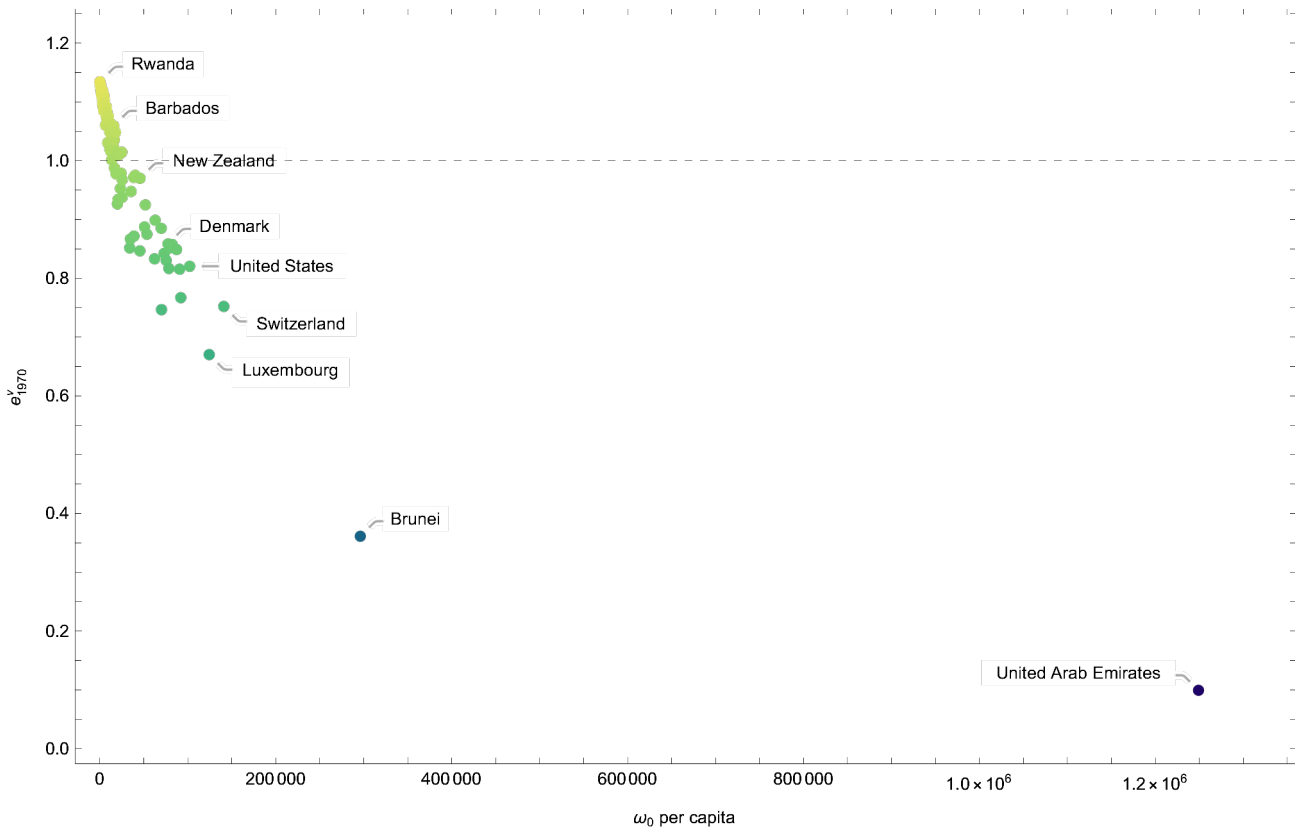
Out[•]= ./ExploitIncomeTable1970.csv

In[•]:= **DataNoHeader[[1]]**

Out[•]= {Rwanda, 684.833, 1.0972, 3.7574, 182.262}

In[∘]:=
```
ExploitWealthPlot = ListPlot[Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, 1, N}] → CountryList,
  LabelingFunction → Callout[Automatic, Automatic], Frame → True, FrameLabel → {"ω0 per capita", "e^v_1970"},
  ColorFunction → "BlueGreenYellow", PlotRange → All, Epilog → {Black, Dashed, Line[{{0, 1}, {2 000 000, 1}}]}]
Export["./ExploitWealthPlot1970.eps", ExploitWealthPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploitWealthPlot1970.eps

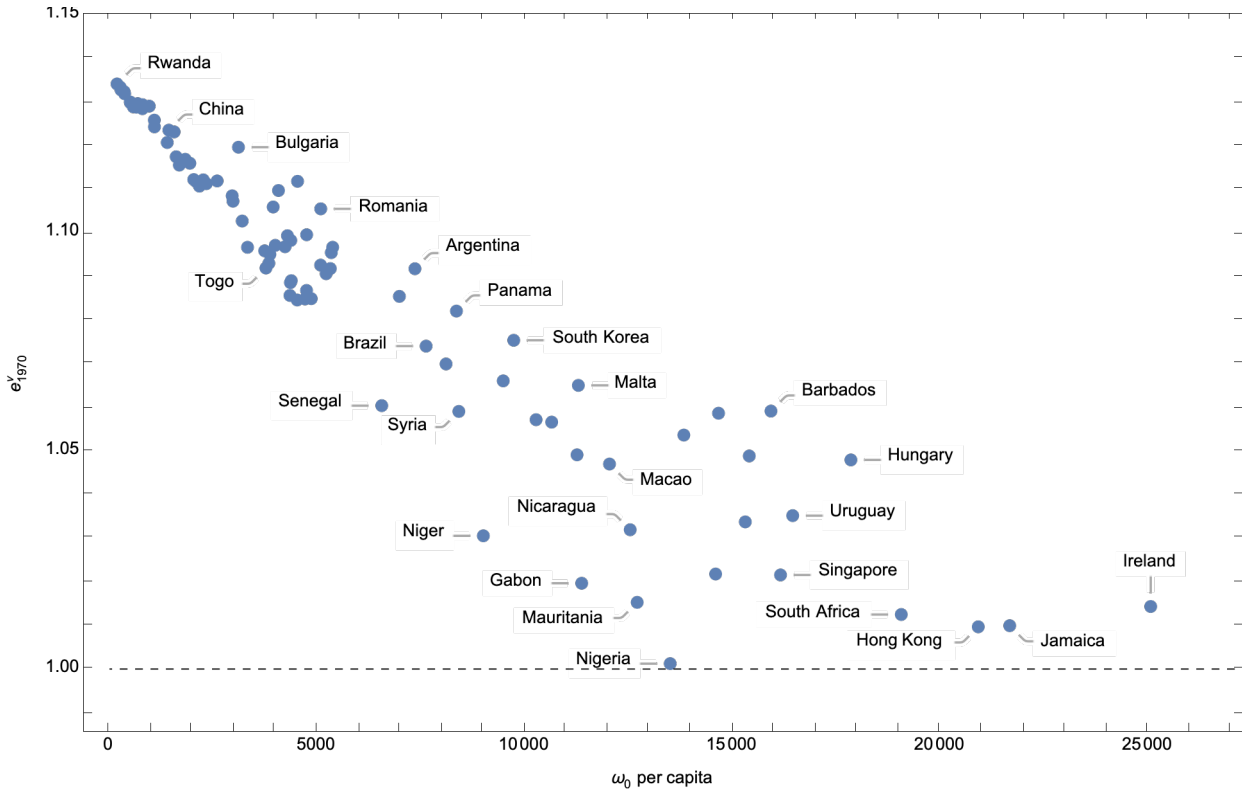
In[ ]:=
```
ExploitedCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1, i, 0.], {i, 1, N}], 0.]
ExploiterCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1, i, 0.], {i, 1, N}], 0.]
```

Out[ ]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32,
 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61,
 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 87, 89, 92, 93, 96}
```

Out[ ]=
```
{86, 88, 90, 91, 94, 95, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109,
 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128}
```

In[ ]:=
```
ExploitedCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploitedCountries}] → CountryList[[ExploitedCountries]],
  PlotRange → All, LabelingFunction → Callout[Automatic, Automatic], Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_1970"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitedCountriesPlot1970.eps", ExploitedCountriesPlot, "EPS"]
```

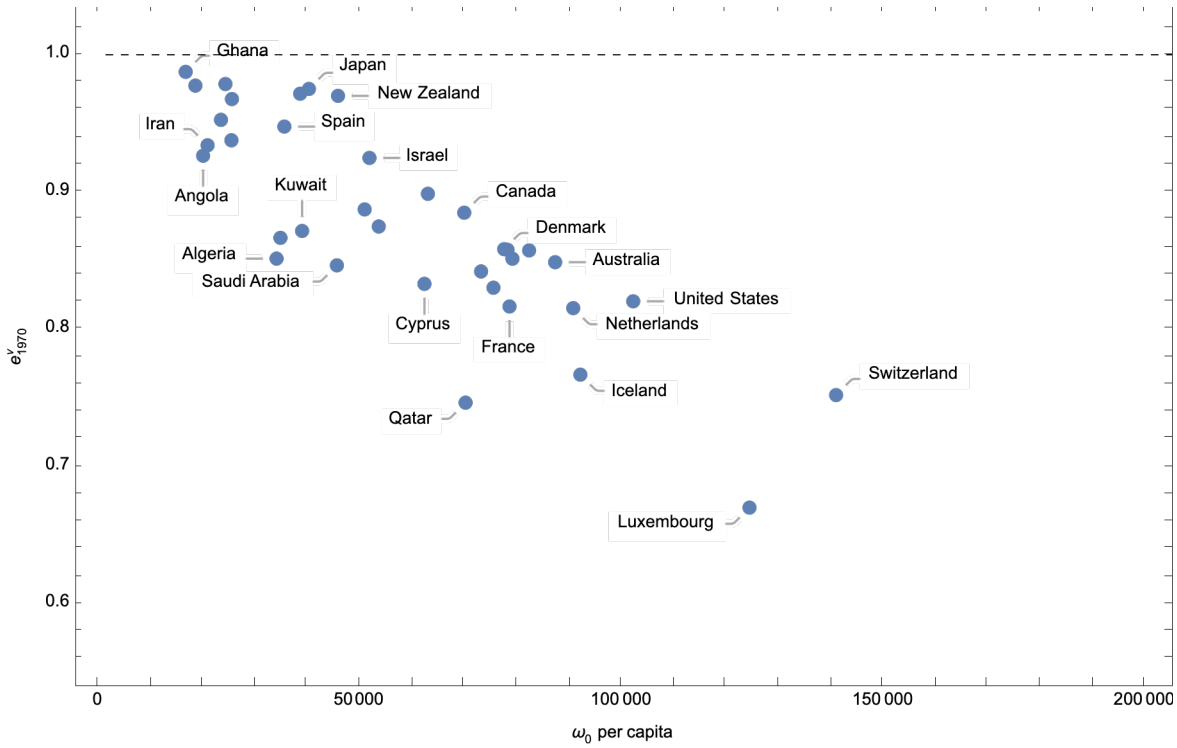
Out[ ]=



Out[ ]= ./ExploitedCountriesPlot1970.eps

In[∘]:=
```
ExploiterCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploiterCountries}] → CountryList[[ExploiterCountries]],
  LabelingFunction → Callout[Automatic, Automatic], Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_1970"}, Epilog → {Black, Dashed, Line[{{0, 1}, {2 000 000, 1}}]}]
Export["./ExploiterCountriesPlot1970.eps", ExploiterCountriesPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploiterCountriesPlot1970.eps

In[∘]:=
```
(outlierTesting = Table[{CountryList[[i]], AgentSet[[i, 1]], AgentSet[[i, 2]]}, {i, 1, Length[CountryList]}]) // TableForm
Export["./outlierTesting1970.csv", outlierTesting, "CSV"];
```

Out[∘]//TableForm=

Rwanda	684.833	412 262.
Myanmar	7095.89	3.16867×10^6

Mali	1655.69	610 784.
Egypt	12 224.	4.05055×10^6
Sierra Leone	1005.13	296 696.
Nepal	6054.72	1.24815×10^6
Ethiopia	14 377.2	2.88754×10^6
Burkina Faso	3240.48	566 622.
Mozambique	5855.74	1.01505×10^6
Malawi	3201.31	624 194.
El Salvador	2914.64	483 306.
Vietnam	34 543.2	6.48555×10^6
Maldives	110.866	19 855.3
Eswatini	465.919	57 699.3
Bangladesh	69 561.5	7.50107×10^6
Uganda	13 077.	1.08249×10^6
China	1.18378×10^6	1.2001×10^8
Lesotho	1602.05	157 483.
Burundi	5587.79	381 135.
Ivory Coast	8578.77	530 669.
Laos	4901.46	324 194.
Madagascar	12 725.1	803 690.
Morocco	32 474.2	1.72659×10^6
Benin	6074.75	316 771.
Sudan	22 263.3	1.11269×10^6
Botswana	1417.23	75 129.4
Pakistan	135 526.	6.92938×10^6
Zimbabwe	13 707.3	719 239.
Indonesia	340 645.	1.56308×10^7
Tanzania	40 235.3	1.76625×10^6
Bulgaria	26 409.5	2.04233×10^6
Cameroon	20 824.8	787 135.
Gambia	1542.92	49 184.6
Congo – Brazzaville	4956.36	154 717.
Togo	7965.76	225 724.
Congo – Kinshasa	76 757.2	2.23243×10^6

India	2.14358×10^6	6.55513×10^7
Paraguay	9748.87	408 713.
Kenya	45 093.	1.45438×10^6
Sri Lanka	50 783.8	2.44021×10^6
Jordan	7272.82	232 651.
Honduras	11 635.7	398 540.
Central African Republic	7866.52	194 150.
Tunisia	22 049.4	579 349.
Mongolia	5579.42	185 503.
Haiti	20 457.7	542 720.
Iraq	44 804.8	1.08255×10^6
Belize	553.078	28 915.
Liberia	6590.44	159 974.
Guatemala	26 660.1	673 885.
Peru	63 862.8	2.20122×10^6
Cambodia	33 972.1	825 949.
Zambia	21 232.5	611 277.
Romania	104 467.	4.31815×10^6
Thailand	192 298.	5.28776×10^6
Mauritius	4391.9	124 092.
Bolivia	23 934.1	739 412.
Fiji	2795.82	89 137.1
Senegal	27 892.8	448 655.
Philippines	249 763.	6.1371×10^6
Argentina	175 525.	4.9557×10^6
Brazil	724 635.	1.43925×10^7
Dominican Republic	36 440.3	676 575.
Panama	12 684.3	291 417.
Syria	53 382.	842 838.
Niger	40 577.1	455 345.
Taiwan	138 159.	2.41478×10^6
South Korea	313 159.	6.3638×10^6
Namibia	8391.25	129 083.
Costa Rica	19 656.7	300 110.

Albania	24 195.7	335 251.
Malta	3615.93	62 221.9
Gabon	6695.38	67 404.1
Macao	2962.01	39 990.3
Nicaragua	30 149.6	343 287.
Mauritania	14 560.5	140 678.
Nigeria	755 270.	6.44746×10^6
Trinidad and Tobago	13 065.1	191 774.
Malaysia	157 570.	1.61899×10^6
Poland	478 305.	7.50867×10^6
Ecuador	92 849.8	1.07762×10^6
Chile	150 597.	2.07989×10^6
Barbados	3802.22	60 100.1
Singapore	33 458.1	343 049.
Uruguay	46 188.9	544 455.
Ghana	145 399.	1.11116×10^6
Hungary	184 918.	2.5256×10^6
Turkey	645 159.	4.57298×10^6
South Africa	420 443.	3.95976×10^6
Angola	117 665.	598 288.
Iran	593 994.	3.15934×10^6
Hong Kong	80 453.5	738 758.
Jamaica	40 624.2	373 901.
Portugal	202 332.	1.20807×10^6
Mexico	1.24804×10^6	8.9223×10^6
Ireland	72 858.6	697 783.
Venezuela	289 324.	1.57316×10^6
Colombia	547 307.	3.61789×10^6
Algeria	491 990.	1.69429×10^6
Bahrain	7387.42	27 344.6
Spain	1.20277×10^6	6.9587×10^6
Austria	289 522.	1.96669×10^6
Kuwait	28 951.6	109 836.

Japan	4.21755×10^6	2.9375×10^7
Saudi Arabia	265 771.	894 545.
New Zealand	128 958.	866 489.
Greece	440 469.	1.80818×10^6
Israel	145 633.	733 588.
Italy	2 865 404	1.10469×10^7
Cyprus	38 208.2	120 934.
United Kingdom	3.49714×10^6	1.52355×10^7
Canada	1.4935×10^6	6.05598×10^6
Qatar	7679.15	16 946.2
Finland	337 144.	1.11191×10^6
Belgium	726 826.	2.27175×10^6
Denmark	382 183.	1.35963×10^6
Norway	302 917.	1.07531×10^6
France	4 080 021	1.20064×10^7
Sweden	637 021.	2.19295×10^6
Germany	6 464 985	2.29001×10^7
Australia	1 116 206	3.79747×10^6
Netherlands	1 179 288	3.45239×10^6
Iceland	18 818.9	45 021.1
United States	21 404 806	6.40398×10^7
Luxembourg	42 253.9	69 981.6
Switzerland	866 890.	1.955×10^6
Brunei	38 352.	20 589.2
United Arab Emirates	292 894.	32 283.6

1980

1980 - Model Setup

1980 - Basic Model

In[•]:=
```
Clear[Activities, i, Wage, Profit, u];
(Activities = Table[Table[{0.0, 0.0, 0.0, 0.0, 0.0, 0.0}, {N}], {T}]);
Wage = Table[0.0, {T}];
Profit = Table[0.0, {T}];

(* First Stage t=1 simulation *)
(* Setting w1 and π1 *)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]], Wage[[1]] = b, Wage[[1]] = RandomReal[{b, (1 - A (1 + 0))/L}]],
  Wage[[1]] = (1 - A (1 + 0))/L];
Profit[[1]] = ((p - p0 A - Wage[[1]] L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^ν,y_t^ν,z_t^ν,δ_t^ν) for all ν and t=1*)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]],
   CapitalConstrained[1, A, L, l, 1, Profit[[1]], b], KnifeEdge[1, A, L, l, 1, Profit[[1]], b]],
  LabourConstrained[1, A, L, l, 1, Profit[[1]], b]
 ];

(* Simulation for remaining T-1 time steps *)
For[t = 2, t ≤ T, t++,

 (* Updating w_t *)
```

```
If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧], If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
   Wage⟦t⟧ = b, Wage⟦t⟧ = RandomReal[{b, ((1 - A (1 + r))/L /. r → 0)}]], Wage⟦t⟧ = ((1 - A (1 + r))/L /. r → 0)];

(* Updating r_t *)
Profit⟦t⟧ = ((p - p0 A - Wage⟦t⟧ L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^v, y_t^v, z_t^v, δ_t^v) for all v *)
If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧],
 If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
  CapitalConstrained[1, A, L, l, t, Profit⟦t⟧, b], KnifeEdge[1, A, L, l, t, Profit⟦t⟧, b]],
 LabourConstrained[1, A, L, l, t, Profit⟦t⟧, b]
];

]
```

Exploitation and Class Over Time

The chart below captures the number of agents who are exploiters, exploited, or neither over the course of the simulation according to Definition 2, which defines exploitation in relation to Λ_t^v and $v_t\, c_t^v$, where v_t is just the embodied labour value and $c_t^v = \pi_t\, W_{t-1}^v + (w_t - b_t)\; l^v + b_t\, \Lambda_t^v$. An agent v is exploited during t if and only if $\Lambda_t^v > v_t\, c_t^v$, an agent is an exploiter if and only if $\Lambda_t^v < v_t\, c_t^v$, and an agent is neither exploited nor an exploiter if and only if $\Lambda_t^v = v_t\, c_t^v$.

In[∘]:=
```
Clear[IndivExploitation, ConsumptionBundle];
ConsumptionBundle = Table[0.0, {i, T}, {j, N}];
IndivExploitation = Table[0.0, {i, T}, {j, N}];

For[i = 1, i ≤ N, i++,
 ConsumptionBundle⟦1, i⟧ = EmbodiedValues⟦1, 2⟧ (Profit⟦1⟧ × AgentSet⟦i, 1⟧ + (Wage⟦1⟧ - b) l⟦i⟧ + b Activities⟦1, i, 6⟧);
```

```
 IndivExploitation⟦1, i⟧ = Activities⟦1, i, 6⟧ - ConsumptionBundle⟦1, i⟧;

]

For[t = 2, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ConsumptionBundle⟦t, i⟧ =
   EmbodiedValues⟦1, 2⟧ (Profit⟦t⟧ × Activities⟦t - 1, i, 5⟧ + (Wage⟦t⟧ - b) l⟦i⟧ + b Activities⟦t, i, 6⟧);
  IndivExploitation⟦t, i⟧ = Activities⟦t, i, 6⟧ - ConsumptionBundle⟦t, i⟧;
 ]
]

Clear[Exploiters, Exploited, NonExploit];
Exploiters = Table[0.0, {T}];
Exploited = Table[0.0, {T}];
NonExploit = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  If[IndivExploitation⟦t, i⟧ < 0, Exploiters⟦t⟧ ++, If[IndivExploitation⟦t, i⟧ == 0, NonExploit⟦t⟧ ++, Exploited⟦t⟧ ++]];
 ]
]

ExploitationLegend =
  Grid[{{Row[{Graphics[{Thick, Blue, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploiters", 20], FontFamily → "Times"]}], Spacer[10],
      Row[{Graphics[{Thick, Red, Dotted, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Neither Exploiter or Exploited", 20], FontFamily → "Times"]}]}},
     {Row[{Graphics[{Thick, Black, Dashed, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploited", 20], FontFamily → "Times"]}], Spacer[10],
     }}, Frame → True, Alignment → Left];
(* ExploitationPlot=Labeled[
```

```
  ListLinePlot[{Exploiters,Exploited,NonExploit},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black},{Thick,Dotted,Red}},
   Frame→True,FrameLabel→{"t","Total ν in Group"},LabelStyle→20,
   PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],ExploitationLegend] *)
```

In[∘]:=
```
Export["./ExploitationPlot.eps", ExploitationPlot, "EPS"]
```

Out[∘]=
```
./ExploitationPlot.eps
```

The chart below captures the class composition of the simulation over time according to Corollary 1 of Theorem 3 (class is defined using labour endowments l^v in relation to means of production).

In[∘]:=
```
Clear[Class1, Class2, Class3, Class4, j, k];
Class1 = Table[0.0, {T}];
Class2 = Table[0.0, {T}];
Class3 = Table[0.0, {T}];
Class4 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ == Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class2⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class3⟦j⟧++, 0];
```

```
 ]
]

Clear[k];
For[k = 1, k ≤ N, k++,
 If[AgentSet⟦k, 1⟧ == 0 && Profit⟦1⟧ > 0, Class4⟦1⟧ ++, 0];
]

Clear[j, k];
For[j = 2, j ≤ T, j ++,
 For[k = 1, k ≤ N, k++,
  If[Activities⟦j - 1, k, 5⟧ == 0 && Profit⟦j⟧ > 0, Class4⟦j⟧ ++, 0];
 ]
]


(* ClassLegend=
 Column[{
   Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^1 = {∑v ∈ (+,0,+)\(+,0,0); A_t y_t^v < z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,DotDashed,Green,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^2 = {∑v ∈ (+,0,0); A_t y_t^v = z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Purple,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^3 = {∑v ∈ (+,+,0)\(+,0,0); A_t y_t^v > z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Black,Dotted,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^4 = {∑v ∈ (0,+,0); W_{t-1}^v = 0}",20],FontFamily→"Times"]}]
  },Frame→True,Alignment→Left];
ClassPlotCorollary1=Labeled[
  ListLinePlot[{Class1,Class2,Class3,Class4},PlotStyle→{{Thick,Blue},{Thick,DotDashed,Green},
     {Thick,Dashed,Purple},{Thick, Dotted,Black},{Thick,DotDashed,Orange},{Thick,Dashed,Green}},
```

```
   PlotRange→{{1,T},{-10,N+10}},Frame→True,FrameLabel→{"t","Total ν in Classes"},
   LabelStyle→20,ImageSize→{500,350},AxesOrigin→{1,-10}],ClassLegend] *)
```

In[∘]:=
```
(* Export["./ClassPlotCorollary1.eps",ClassPlotCorollary1,"EPS"] *)
```

The chart below captures the intersection of class and exploitation over the simulation. If an agent is in C^1 according to Corollary 1 of Theorem 3 and is also an exploiter according to Definition 2, then the agent is in the group "$C^1 \wedge$ Exploiter". If an agent is in $C^3 \cup C^4$ and is exploited they are in group "$(C^3 \cup C^4) \wedge$ Exploited".

In[∘]:=
```
Clear[CECP1, CECP2, CECP3, j, k];
CECP1 = Table[0.0, {T}];
CECP2 = Table[0.0, {T}];
CECP3a = Table[0.0, {T}];
CECP3b = Table[0.0, {T}];
CECP3 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ < 0, CECP1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
  If[If[j == 1, AgentSet⟦k, 1⟧, Activities⟦j - 1, k, 5⟧] == 0 && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
 ]
]


(* CECPLegend=
 Grid[{{Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],Spacer[5],
```

```
      Style[Style["∑v ∈ C_t^1 ∧ Exploiter",20],FontFamily→"Times"]}],Spacer[10],
    Row[{Graphics[{Thick,Black,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
      Spacer[5],Style[Style["∑v ∈ (C_t^3 ⋃ C_t^4) ∧ Exploited",20],FontFamily→"Times"]}],
   }},Frame→True,Alignment→Left];
CECPPlotCorollary1=
 Labeled[ListLinePlot[{CECP1,CECP3},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black}},Frame→True,FrameLabel→
    {"t","Total v in Group"},LabelStyle→20,PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],CECPLegend] *)
```

In[∘]:=
```
(* Export["./CECPPlotCorollary1.eps",CECPPlotCorollary1,"EPS"] *)
```

The charts below display the distribution of an index of the intensity of exploitation across the agents over the simulation. The index itself is calculated as $e_t^v = \Lambda_t^v / v_t\, c_t^v$ and the charts that follow explore some possibilities for visualizing the intensity of exploitation over time.

In[∘]:=
```
Clear[ExploitationIndex];
ExploitationIndex = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndex[[t, i]] = Activities[[t, i, 6]] / (EmbodiedValues[[1, 2]] (ConsumptionBundle[[t, i]] / EmbodiedValues[[1, 2]])) /. p → 1
 ]
]
```

The chart below uses *Mathematica's* `ArrayPlot` function to display the exploitation index of the N agents over *T*. There is a fixed distribution of uneven exploitation intensity while the simulation is capital constrained, however, this pattern disappears once the simulation is labour constrained, at which point exploitation intensity is the same for all $v \in \mathcal{N}$.

In[∘]:=
```
ExploitationIndex[[1, 1]]
```

Out[∘]=
```
1.13361
```

In[∘]:=
```
ExploitationIndex[[1]]
```

Out[∘]=
```
{1.13361, 1.13317, 1.13231, 1.13143, 1.13048, 1.13035, 1.12665, 1.13027, 1.126, 1.12408, 1.12695, 1.12527,
 1.12449, 1.12457, 1.12624, 1.12141, 1.1187, 1.12131, 1.12156, 1.12026, 1.11003, 1.11844, 1.11283, 1.10667,
 1.10707, 1.1089, 1.10568, 1.11238, 1.09857, 1.09909, 1.1075, 1.10073, 1.09775, 1.09994, 1.10285, 1.09832,
 1.11197, 1.09888, 1.10144, 1.08802, 1.08335, 1.09936, 1.08831, 1.10323, 1.10395, 1.09047, 1.09429, 1.1103,
 1.10174, 1.10629, 1.08554, 1.07774, 1.10019, 1.07918, 1.09118, 1.08512, 1.05446, 1.08716, 1.07253, 1.08597,
 1.05394, 1.08883, 1.06042, 1.08328, 1.04121, 1.06656, 1.03751, 1.05397, 1.07022, 1.03628, 1.05501, 1.04544,
 0.990084, 1.05517, 1.05135, 1.03457, 1.01613, 1.03384, 0.966249, 0.942564, 1.02927, 1.00707, 1.02482, 1.0109,
 0.999821, 0.999544, 1.01228, 0.939594, 0.947837, 0.995262, 0.904061, 0.999001, 0.966151, 1.00707, 0.928616,
 0.926858, 0.912327, 0.834185, 0.964483, 0.8875, 0.959635, 0.947002, 0.930484, 0.924232, 0.855113, 0.76213,
 0.768074, 0.828668, 0.816834, 0.871962, 0.875636, 0.866202, 0.727924, 0.817254, 0.845513, 0.838071, 0.844042,
 0.765525, 0.858519, 0.806109, 0.787134, 0.821024, 0.827708, 0.715438, 0.693749, 0.741867, 0.585412, 0.217234}
```

In[∘]:=
```
CountryList
```

Out[∘]=
```
{Rwanda, Myanmar, Sierra Leone, Mali, Ethiopia, Egypt, Nepal, Vietnam, Mozambique, Burkina Faso, El Salvador,
 Malawi, Uganda, Bangladesh, Maldives, Madagascar, Burundi, Laos, Zimbabwe, China, Benin, Lesotho, Pakistan,
 Ivory Coast, Sudan, India, Morocco, Eswatini, Gambia, Central African Republic, Indonesia, Cameroon,
 Liberia, Tanzania, Kenya, Cambodia, Sri Lanka, Congo - Brazzaville, Zambia, Congo - Kinshasa, Senegal,
 Honduras, Togo, Bolivia, Peru, Botswana, Syria, Bulgaria, Philippines, Belize, Guatemala, Haiti, Mongolia,
 Tunisia, Mauritius, Thailand, Niger, Paraguay, Jordan, Fiji, Iraq, Argentina, Dominican Republic, Romania,
 Brazil, Panama, Ghana, Albania, South Korea, Nicaragua, Costa Rica, Namibia, Mauritania, Poland, Chile,
 Ecuador, Malaysia, Jamaica, Iran, Angola, Malta, Colombia, Barbados, Taiwan, South Africa, Mexico,
 Trinidad and Tobago, Gabon, Turkey, Uruguay, Nigeria, Hong Kong, Macao, Hungary, Singapore, Portugal,
 Venezuela, Algeria, Ireland, Bahrain, New Zealand, United Kingdom, Austria, Israel, Spain, Saudi Arabia,
 Kuwait, Italy, Greece, Japan, Canada, Germany, Qatar, Belgium, Denmark, Netherlands, Sweden, Cyprus, Australia,
 Finland, France, Norway, United States, Iceland, Luxembourg, Switzerland, Brunei, United Arab Emirates}
```

In[∘]:=
```
(ExploitTable = Table[{CountryList[[i]], ExploitationIndex[[1, i]], DataNoHeader[[i, 5]]}, {i, 1, N}]) // TableForm
Export["./ExploitTable1980.csv", ExploitTable, "CSV"]
```

Out[∘]//TableForm=

Rwanda	1.13361	289.35
Myanmar	1.13317	357.518

Sierra Leone	1.13231	412.043
Mali	1.13143	464.601
Ethiopia	1.13048	542.742
Egypt	1.13035	689.718
Nepal	1.12665	949.934
Vietnam	1.13027	950.097
Mozambique	1.126	1027.49
Burkina Faso	1.12408	1122.24
El Salvador	1.12695	1202.27
Malawi	1.12527	1337.8
Uganda	1.12449	1364.65
Bangladesh	1.12457	1383.17
Maldives	1.12624	1631.93
Madagascar	1.12141	1679.46
Burundi	1.1187	1753.54
Laos	1.12131	1809.73
Zimbabwe	1.12156	2061.72
China	1.12026	2528.7
Benin	1.11003	2620.45
Lesotho	1.11844	2660.82
Pakistan	1.11283	2706.85
Ivory Coast	1.10667	3001.14
Sudan	1.10707	3022.99
India	1.1089	3221.81
Morocco	1.10568	3285.57
Eswatini	1.11238	3513.13
Gambia	1.09857	3812.58
Central African Republic	1.09909	3821.95
Indonesia	1.1075	4004.98
Cameroon	1.10073	4241.15
Liberia	1.09775	4374.94
Tanzania	1.09994	4386.04
Kenya	1.10285	4526.79
Cambodia	1.09832	4654.81
Sri Lanka	1.11197	4707.71
Congo – Brazzaville	1.09888	4796.08

Zambia	1.10144	5032.27
Congo – Kinshasa	1.08802	5339.67
Senegal	1.08335	5403.67
Honduras	1.09936	5507.99
Togo	1.08831	5628.35
Bolivia	1.10323	5638.98
Peru	1.10395	5775.78
Botswana	1.09047	5789.95
Syria	1.09429	5864.85
Bulgaria	1.1103	6126.81
Philippines	1.10174	6258.45
Belize	1.10629	6420.13
Guatemala	1.08554	6502.52
Haiti	1.07774	6645.81
Mongolia	1.10019	6851.63
Tunisia	1.07918	7035.68
Mauritius	1.09118	7363.6
Thailand	1.08512	7459.93
Niger	1.05446	8021.61
Paraguay	1.08716	8352.77
Jordan	1.07253	9078.39
Fiji	1.08597	9082.58
Iraq	1.05394	9826.1
Argentina	1.08883	9943.16
Dominican Republic	1.06042	12 340.7
Romania	1.08328	12 351.8
Brazil	1.04121	13 706.1
Panama	1.06656	13 932.9
Ghana	1.03751	14 575.9
Albania	1.05397	14 886.5
South Korea	1.07022	14 985.8
Nicaragua	1.03628	15 034.8
Costa Rica	1.05501	15 234.2
Namibia	1.04544	15 453.5
Mauritania	0.990084	19 013.2
Poland	1.05517	19 444.

Chile	1.05135	19 528.2
Ecuador	1.03457	19 657.5
Malaysia	1.01613	21 332.1
Jamaica	1.03384	21 460.1
Iran	0.966249	21 583.7
Angola	0.942564	21 670.2
Malta	1.02927	22 889.6
Colombia	1.00707	23 248.7
Barbados	1.02482	23 732.7
Taiwan	1.0109	24 237.2
South Africa	0.999821	25 802.1
Mexico	0.999544	26 598.9
Trinidad and Tobago	1.01228	28 663.8
Gabon	0.939594	29 074.4
Turkey	0.947837	29 579.3
Uruguay	0.995262	30 785.8
Nigeria	0.904061	31 046.2
Hong Kong	0.999001	31 359.9
Macao	0.966151	32 606.1
Hungary	1.00707	34 429.4
Singapore	0.928616	37 389.4
Portugal	0.926858	37 651.5
Venezuela	0.912327	44 352.3
Algeria	0.834185	45 931.1
Ireland	0.964483	46 683.
Bahrain	0.8875	49 118.3
New Zealand	0.959635	59 760.2
United Kingdom	0.947002	60 167.4
Austria	0.930484	62 552.1
Israel	0.924232	67 815.1
Spain	0.855113	73 455.5
Saudi Arabia	0.76213	85 177.9
Kuwait	0.768074	85 365.6
Italy	0.828668	86 133.3
Greece	0.816834	90 480.3
Japan	0.871962	92 250.3

```
Canada                    0.875636   93 010.5
Germany                   0.866202   100 359.
Qatar                     0.727924   101 183.
Belgium                   0.817254   101 529.
Denmark                   0.845513   102 487.
Netherlands               0.838071   102 598.
Sweden                    0.844042   102 922.
Cyprus                    0.765525   104 044.
Australia                 0.858519   105 280.
Finland                   0.806109   112 562.
France                    0.787134   116 173.
Norway                    0.821024   116 907.
United States             0.827708   126 303.
Iceland                   0.715438   144 555.
Luxembourg                0.693749   167 884.
Switzerland               0.741867   179 085.
Brunei                    0.585412   201 402.
United Arab Emirates      0.217234   692 824.
```

Out[•]= ./ExploitTable1980.csv

In[•]:=
```
(* ExploitationArray=ArrayPlot[Transpose[ExploitationIndex],
  FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./ExploitationArray.eps",ExploitationArray,"EPS"] *)
```

In[•]:=
```
ExploitationIndexDistrib = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndexDistrib[[t, i]] = {t, ExploitationIndex[[t, i]]}
 ]
]
```

Below the Gini coefficient for the exploitation intensity index is plotted over *T*. The pattern of the Gini coefficient is consistent with the previous charts depicting the exploitation intensity index.

In[∘]:=
```
(* Gini=Table[0.0,{T}];
For[t=1,t≤T,t++,
 Gini[[t]]=N/(N-1) ((Σ_{i=1}^N Σ_{j=1}^N Abs[ExploitationIndex[[t,i]]-ExploitationIndex[[t,j]]])/(2 N^2 Mean[ExploitationIndex[[t]]]))
] *)
```

In[∘]:=
```
Clear[SortExploitationIndex];
SortExploitationIndex = Table[Table[0.0, {N}], {t, 1, T}];
GiniTest = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortExploitationIndex[[t]] = Sort[ExploitationIndex[[t]]];
 GiniTest[[t]] = N/(N - 1) Σ_{i=1}^N ((2 i - N - 1) SortExploitationIndex[[t, i]]) / (N^2 Mean[SortExploitationIndex[[t]]])
]
```

In[∘]:=
```
Chop[GiniTest]
```

Out[∘]= {0.0698046}

In[∘]:=
```
(* ExploitationGini=ListLinePlot[GiniTest,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: e_t^y"}] *)
```

In[∘]:=
```
(* Export["./ExploitationGini.eps",ExploitationGini,"EPS"] *)
```

Plotting Gini coefficient for wealth $W_{t-1} = p_{t-1}\,\omega_{t-1}$.

```
In[•]:= Clear[SortWealth, WealthGini];
SortWealth = Table[Table[0.0, {N}], {t, 1, T}];
WealthGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortWealth[[t]] = If[t == 1, Sort[AgentSet[[All, 1]]], Sort[Activities[[t - 1, All, 5]]]];
 WealthGini[[t]] = N/(N - 1) Sum[(2 i - N - 1) SortWealth[[t, i]]/(N^2 Mean[SortWealth[[t]]]), {i, 1, N}]
]
(* WealthGiniPlot=
 ListLinePlot[WealthGini,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: W^γ_{t-1}"},LabelStyle→12,PlotRange→{0,1}] *)

In[•]:= (* Export["./WealthGini.eps",WealthGiniPlot,"EPS"] *)

In[•]:= WealthGini

Out[•]= {0.855665}
```

The figure below plots the distribution of wealth for select *t*.

```
In[•]:= (* WealthDistribRow=GraphicsRow[{
    Histogram[AgentSet[[All,1]],10,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 1",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[24,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 25",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[49,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 50",18],AxesLabel→{"ω^γ_{t-1}"}]
   },Spacings→{15,-20}
  ]
  Export["./WealthDistribRow.eps",WealthDistribRow,"EPS"]
*)
```

Income Figures

Figures on net and gross income.

In[∘]:=
```
Clear[Income];
Income = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  Income⟦t, i⟧ = Profit⟦t⟧ × Activities⟦t, i, 5⟧ + Wage⟦t⟧ × Activities⟦t, i, 6⟧
 ]
]

IncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
```

$$\text{IncomeGini}[[t]] = \frac{N}{N-1}\left(\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\text{Abs}[\text{Income}[[t, i]] - \text{Income}[[t, j]]]\right) \Big/ \left(2\,N^2\,\text{Mean}[\text{Income}[[t]]]\right)\right)$$

```
]
(* IncomeGiniPlot=ListLinePlot[IncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[IncomeGini]+0.1},AxesLabel→{"t","Gini: Net Income"},LabelStyle→14] *)
```

In[∘]:=
```
(* Export["./NetIncomeGini.eps",IncomeGiniPlot,"EPS"] *)
```

```
In[•]:= Clear[IncomeShares];
IncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  IncomeShares[[t, i]] = Income[[t, i]] / Total[Income[[t]]]
 ]
]
(* IncomeArray=ArrayPlot[Transpose[IncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},
  PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",ColorFunctionScaling→True,
  DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./NetIncomeArray.eps",IncomeArray,"EPS"] *)

In[•]:= Clear[GrossIncome];
GrossIncome = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncome[[t, i]] = (1 + Profit[[t]]) Activities[[t, i, 5]] + Wage[[t]] × Activities[[t, i, 6]]
 ]
]

GrossIncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 GrossIncomeGini[[t]] = N/(N - 1) ((Sum[Sum[Abs[GrossIncome[[t, i]] - GrossIncome[[t, j]]], {j, 1, N}], {i, 1, N}]) / (2 N^2 Mean[GrossIncome[[t]]]))
]
(* GrossIncomeGiniPlot=ListLinePlot[GrossIncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[GrossIncomeGini]+0.1},AxesLabel→{"t","Gini: Income"},LabelStyle→14] *)
```

```
In[•]:= (* Export["./GrossIncomeGini.eps",GrossIncomeGiniPlot,"EPS"] *)

In[•]:= GrossIncomeGini

Out[•]= {0.830136}

In[•]:= Clear[GrossIncomeShares];
GrossIncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncomeShares[[t, i]] = GrossIncome[[t, i]] / Total[GrossIncome[[t]]]
 ]
]
(* GrossIncomeArray=
 ArrayPlot[Transpose[GrossIncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./GrossIncomeArray.eps",GrossIncomeArray,"EPS"] *)
```

Updated Simulation Reporting

```
In[•]:= (ExploitedTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploitedTable1980.csv", ExploitedTable, "CSV"]

Out[•]//TableForm=
Rwanda          &   1.13361   \\
Myanmar         &   1.13317   \\
Sierra Leone    &   1.13231   \\
Mali            &   1.13143   \\
Ethiopia        &   1.13048   \\
Egypt           &   1.13035   \\
Nepal           &   1.12665   \\
```

```
Vietnam                    &   1.13027   \\
Mozambique                 &   1.126     \\
Burkina Faso               &   1.12408   \\
El Salvador                &   1.12695   \\
Malawi                     &   1.12527   \\
Uganda                     &   1.12449   \\
Bangladesh                 &   1.12457   \\
Maldives                   &   1.12624   \\
Madagascar                 &   1.12141   \\
Burundi                    &   1.1187    \\
Laos                       &   1.12131   \\
Zimbabwe                   &   1.12156   \\
China                      &   1.12026   \\
Benin                      &   1.11003   \\
Lesotho                    &   1.11844   \\
Pakistan                   &   1.11283   \\
Ivory Coast                &   1.10667   \\
Sudan                      &   1.10707   \\
India                      &   1.1089    \\
Morocco                    &   1.10568   \\
Eswatini                   &   1.11238   \\
Gambia                     &   1.09857   \\
Central African Republic   &   1.09909   \\
Indonesia                  &   1.1075    \\
Cameroon                   &   1.10073   \\
Liberia                    &   1.09775   \\
Tanzania                   &   1.09994   \\
Kenya                      &   1.10285   \\
Cambodia                   &   1.09832   \\
Sri Lanka                  &   1.11197   \\
Congo - Brazzaville        &   1.09888   \\
Zambia                     &   1.10144   \\
Congo - Kinshasa           &   1.08802   \\
Senegal                    &   1.08335   \\
Honduras                   &   1.09936   \\
Togo                       &   1.08831   \\
```

```
Bolivia                  &   1.10323   \\
Peru                     &   1.10395   \\
Botswana                 &   1.09047   \\
Syria                    &   1.09429   \\
Bulgaria                 &   1.1103    \\
Philippines              &   1.10174   \\
Belize                   &   1.10629   \\
Guatemala                &   1.08554   \\
Haiti                    &   1.07774   \\
Mongolia                 &   1.10019   \\
Tunisia                  &   1.07918   \\
Mauritius                &   1.09118   \\
Thailand                 &   1.08512   \\
Niger                    &   1.05446   \\
Paraguay                 &   1.08716   \\
Jordan                   &   1.07253   \\
Fiji                     &   1.08597   \\
Iraq                     &   1.05394   \\
Argentina                &   1.08883   \\
Dominican Republic       &   1.06042   \\
Romania                  &   1.08328   \\
Brazil                   &   1.04121   \\
Panama                   &   1.06656   \\
Ghana                    &   1.03751   \\
Albania                  &   1.05397   \\
South Korea              &   1.07022   \\
Nicaragua                &   1.03628   \\
Costa Rica               &   1.05501   \\
Namibia                  &   1.04544   \\
Poland                   &   1.05517   \\
Chile                    &   1.05135   \\
Ecuador                  &   1.03457   \\
Malaysia                 &   1.01613   \\
Jamaica                  &   1.03384   \\
Malta                    &   1.02927   \\
Colombia                 &   1.00707   \\
```

```
Barbados                    &    1.02482    \\
Taiwan                      &    1.0109     \\
Trinidad and Tobago         &    1.01228    \\
Hungary                     &    1.00707    \\
```

Out[•]= ./ExploitedTable1980.csv

In[•]:=
```
(ExploiterTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploiterTable1980.csv", ExploiterTable, "CSV"]
```

Out[•]//TableForm=
```
Mauritania              &    0.990084    \\
Iran                    &    0.966249    \\
Angola                  &    0.942564    \\
South Africa            &    0.999821    \\
Mexico                  &    0.999544    \\
Gabon                   &    0.939594    \\
Turkey                  &    0.947837    \\
Uruguay                 &    0.995262    \\
Nigeria                 &    0.904061    \\
Hong Kong               &    0.999001    \\
Macao                   &    0.966151    \\
Singapore               &    0.928616    \\
Portugal                &    0.926858    \\
Venezuela               &    0.912327    \\
Algeria                 &    0.834185    \\
Ireland                 &    0.964483    \\
Bahrain                 &    0.8875      \\
New Zealand             &    0.959635    \\
United Kingdom          &    0.947002    \\
Austria                 &    0.930484    \\
Israel                  &    0.924232    \\
Spain                   &    0.855113    \\
Saudi Arabia            &    0.76213     \\
Kuwait                  &    0.768074    \\
Italy                   &    0.828668    \\
Greece                  &    0.816834    \\
```

```
Japan                  &    0.871962    \\
Canada                 &    0.875636    \\
Germany                &    0.866202    \\
Qatar                  &    0.727924    \\
Belgium                &    0.817254    \\
Denmark                &    0.845513    \\
Netherlands            &    0.838071    \\
Sweden                 &    0.844042    \\
Cyprus                 &    0.765525    \\
Australia              &    0.858519    \\
Finland                &    0.806109    \\
France                 &    0.787134    \\
Norway                 &    0.821024    \\
United States          &    0.827708    \\
Iceland                &    0.715438    \\
Luxembourg             &    0.693749    \\
Switzerland            &    0.741867    \\
Brunei                 &    0.585412    \\
United Arab Emirates   &    0.217234    \\
```

Out[•]= ./ExploiterTable1980.csv

In[•]:=
```
(ExploitIncomeCSV = Table[{CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"}, {i, 1, N}]) // TableForm
Export["./ExploitIncomeTable1980.csv", ExploitIncomeCSV, "CSV"]
```

Out[•]//TableForm=
```
Rwanda                     &    1.13361     \\
Myanmar                    &    1.13317     \\
Sierra Leone               &    1.13231     \\
Mali                       &    1.13143     \\
Ethiopia                   &    1.13048     \\
Egypt                      &    1.13035     \\
Nepal                      &    1.12665     \\
Vietnam                    &    1.13027     \\
Mozambique                 &    1.126       \\
Burkina Faso               &    1.12408     \\
El Salvador                &    1.12695     \\
Malawi                     &    1.12527     \\
```

```
Uganda                    &   1.12449    \\
Bangladesh                &   1.12457    \\
Maldives                  &   1.12624    \\
Madagascar                &   1.12141    \\
Burundi                   &   1.1187     \\
Laos                      &   1.12131    \\
Zimbabwe                  &   1.12156    \\
China                     &   1.12026    \\
Benin                     &   1.11003    \\
Lesotho                   &   1.11844    \\
Pakistan                  &   1.11283    \\
Ivory Coast               &   1.10667    \\
Sudan                     &   1.10707    \\
India                     &   1.1089     \\
Morocco                   &   1.10568    \\
Eswatini                  &   1.11238    \\
Gambia                    &   1.09857    \\
Central African Republic  &   1.09909    \\
Indonesia                 &   1.1075     \\
Cameroon                  &   1.10073    \\
Liberia                   &   1.09775    \\
Tanzania                  &   1.09994    \\
Kenya                     &   1.10285    \\
Cambodia                  &   1.09832    \\
Sri Lanka                 &   1.11197    \\
Congo - Brazzaville       &   1.09888    \\
Zambia                    &   1.10144    \\
Congo - Kinshasa          &   1.08802    \\
Senegal                   &   1.08335    \\
Honduras                  &   1.09936    \\
Togo                      &   1.08831    \\
Bolivia                   &   1.10323    \\
Peru                      &   1.10395    \\
Botswana                  &   1.09047    \\
Syria                     &   1.09429    \\
Bulgaria                  &   1.1103     \\
```

```
Philippines                &    1.10174     \\
Belize                     &    1.10629     \\
Guatemala                  &    1.08554     \\
Haiti                      &    1.07774     \\
Mongolia                   &    1.10019     \\
Tunisia                    &    1.07918     \\
Mauritius                  &    1.09118     \\
Thailand                   &    1.08512     \\
Niger                      &    1.05446     \\
Paraguay                   &    1.08716     \\
Jordan                     &    1.07253     \\
Fiji                       &    1.08597     \\
Iraq                       &    1.05394     \\
Argentina                  &    1.08883     \\
Dominican Republic         &    1.06042     \\
Romania                    &    1.08328     \\
Brazil                     &    1.04121     \\
Panama                     &    1.06656     \\
Ghana                      &    1.03751     \\
Albania                    &    1.05397     \\
South Korea                &    1.07022     \\
Nicaragua                  &    1.03628     \\
Costa Rica                 &    1.05501     \\
Namibia                    &    1.04544     \\
Mauritania                 &    0.990084    \\
Poland                     &    1.05517     \\
Chile                      &    1.05135     \\
Ecuador                    &    1.03457     \\
Malaysia                   &    1.01613     \\
Jamaica                    &    1.03384     \\
Iran                       &    0.966249    \\
Angola                     &    0.942564    \\
Malta                      &    1.02927     \\
Colombia                   &    1.00707     \\
Barbados                   &    1.02482     \\
Taiwan                     &    1.0109      \\
```

```
South Africa              &   0.999821   \\
Mexico                    &   0.999544   \\
Trinidad and Tobago       &   1.01228    \\
Gabon                     &   0.939594   \\
Turkey                    &   0.947837   \\
Uruguay                   &   0.995262   \\
Nigeria                   &   0.904061   \\
Hong Kong                 &   0.999001   \\
Macao                     &   0.966151   \\
Hungary                   &   1.00707    \\
Singapore                 &   0.928616   \\
Portugal                  &   0.926858   \\
Venezuela                 &   0.912327   \\
Algeria                   &   0.834185   \\
Ireland                   &   0.964483   \\
Bahrain                   &   0.8875     \\
New Zealand               &   0.959635   \\
United Kingdom            &   0.947002   \\
Austria                   &   0.930484   \\
Israel                    &   0.924232   \\
Spain                     &   0.855113   \\
Saudi Arabia              &   0.76213    \\
Kuwait                    &   0.768074   \\
Italy                     &   0.828668   \\
Greece                    &   0.816834   \\
Japan                     &   0.871962   \\
Canada                    &   0.875636   \\
Germany                   &   0.866202   \\
Qatar                     &   0.727924   \\
Belgium                   &   0.817254   \\
Denmark                   &   0.845513   \\
Netherlands               &   0.838071   \\
Sweden                    &   0.844042   \\
Cyprus                    &   0.765525   \\
Australia                 &   0.858519   \\
Finland                   &   0.806109   \\
```

```
France                    &   0.787134   \\
Norway                    &   0.821024   \\
United States             &   0.827708   \\
Iceland                   &   0.715438   \\
Luxembourg                &   0.693749   \\
Switzerland               &   0.741867   \\
Brunei                    &   0.585412   \\
United Arab Emirates      &   0.217234   \\
```

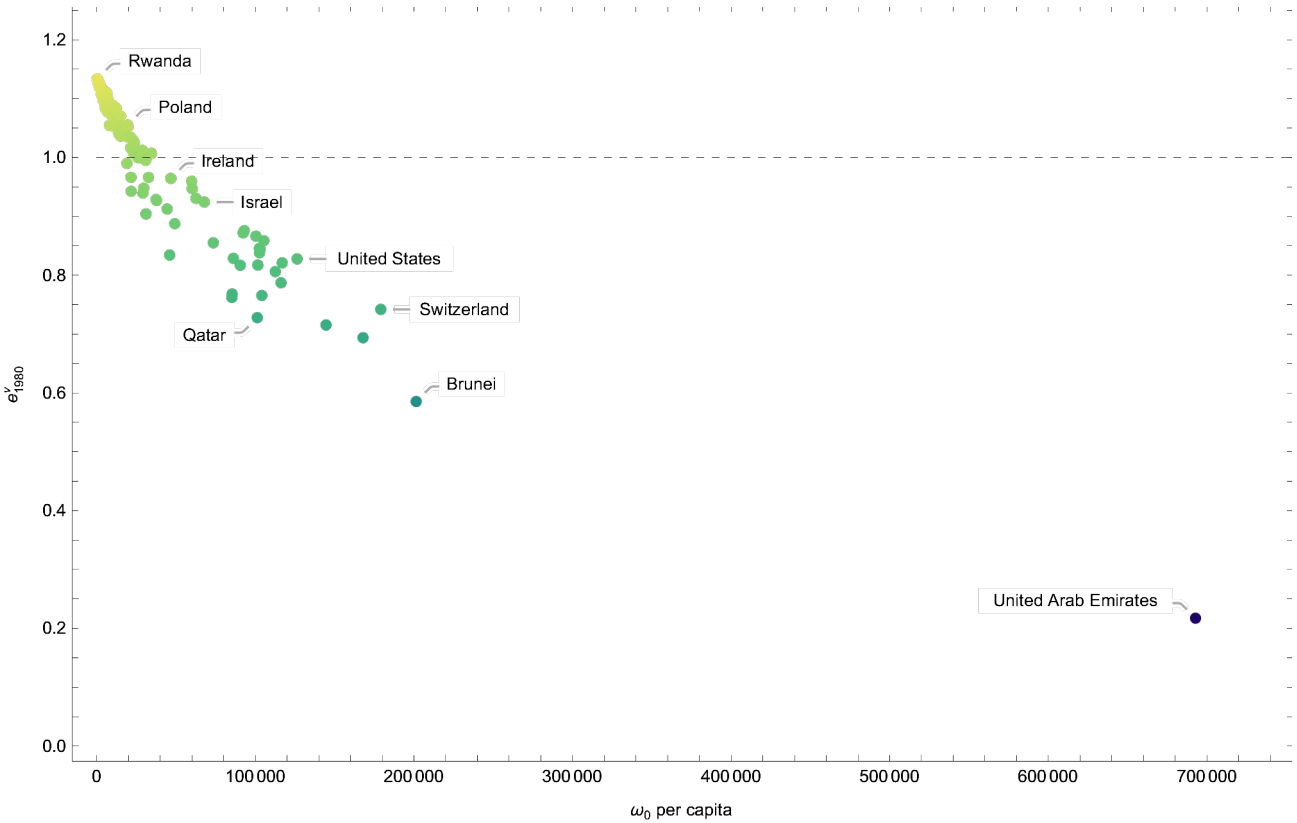
Out[•]= ./ExploitIncomeTable1980.csv

In[•]:= **DataNoHeader[[1]]**

Out[•]= {Rwanda, 1491.11, 1.1763, 5.1533, 289.35}

In[∘]:=
```
ExploitWealthPlot = ListPlot[Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, 1, N}] → CountryList,
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^Y_1980"}, ColorFunction → "BlueGreenYellow",
  Epilog → {Black, Dashed, Line[{{0, 1}, {1 000 000, 1}}]}]
Export["./ExploitWealthPlot1980.eps", ExploitWealthPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploitWealthPlot1980.eps

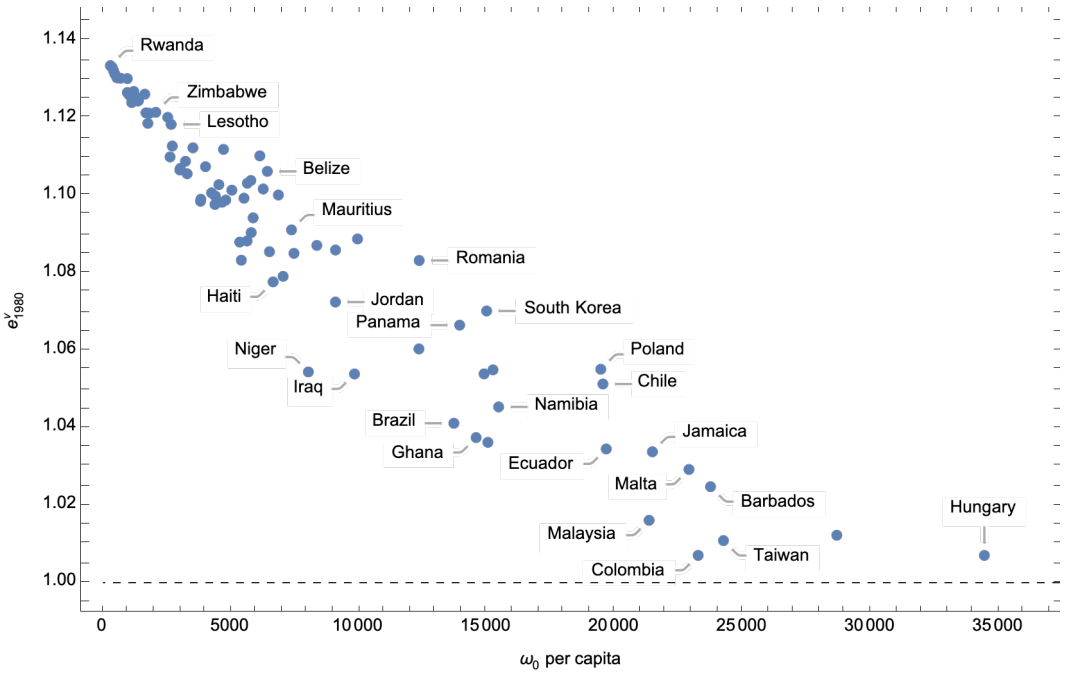
In[∘]:=
```
ExploitedCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1, i, 0.], {i, 1, N}], 0.]
ExploiterCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1, i, 0.], {i, 1, N}], 0.]
```

Out[∘]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29,
 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56,
 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 74, 75, 76, 77, 78, 81, 82, 83, 84, 87, 94}
```

Out[∘]=
```
{73, 79, 80, 85, 86, 88, 89, 90, 91, 92, 93, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128}
```

In[∘]:=
```
ExploitedCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploitedCountries}] → CountryList[[ExploitedCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^γ_1980"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitedCountriesPlot1980.eps", ExploitedCountriesPlot, "EPS"]
```

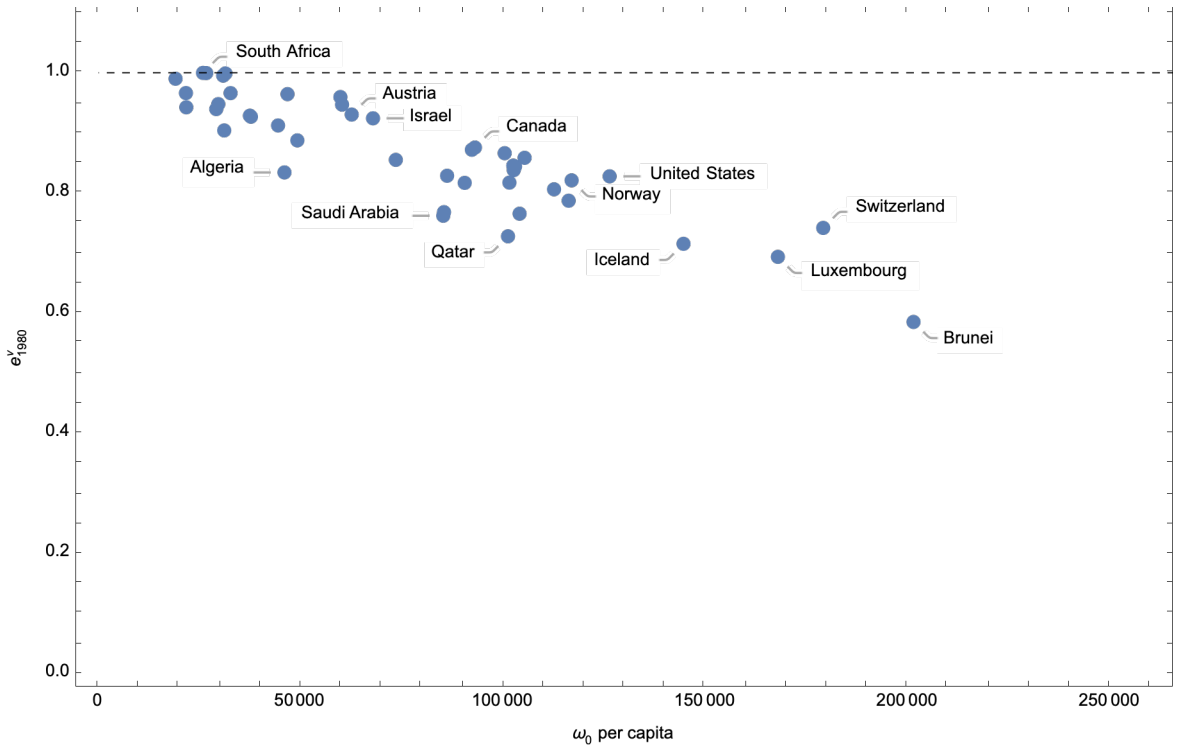
Out[∘]=



Out[∘]= ./ExploitedCountriesPlot1980.eps

In[∘]:=
```
ExploiterCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploiterCountries}] → CountryList[[ExploiterCountries]],
  LabelingFunction → Callout[Automatic, Automatic], Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_1980"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploiterCountriesPlot1980.eps", ExploiterCountriesPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploiterCountriesPlot1980.eps

In[∘]:=
```
(outlierTesting = Table[{CountryList[[i]], AgentSet[[i, 1]], AgentSet[[i, 2]]}, {i, 1, Length[CountryList]}]) // TableForm
Export["./outlierTesting1980.csv", outlierTesting, "CSV"];
```

Out[∘]//TableForm=

Rwanda	1491.11	606 183.
Myanmar	12 235.8	4.28591×10^6

Sierra Leone	1396.21	385 611.
Mali	3294.07	745 595.
Ethiopia	19 072.9	3.61889×10^6
Egypt	29 871.1	5.5427×10^6
Nepal	14 264.6	1.63484×10^6
Vietnam	51 573.	9.44558×10^6
Mozambique	11 949.9	1.28246×10^6
Burkina Faso	7656.79	691 968.
El Salvador	5519.74	652 671.
Malawi	8361.78	838 054.
Uganda	16 979.3	1.58851×10^6
Bangladesh	110 155.	1.03746×10^7
Maldives	258.334	28 375.3
Madagascar	14 639.2	1.08461×10^6
Burundi	7289.99	456 014.
Laos	5896.29	433 816.
Zimbabwe	15 274.5	1.143×10^6
China	2.52892×10^6	1.73755×10^8
Benin	9740.75	405 547.
Lesotho	3566.29	219 863.
Pakistan	211 282.	9.87075×10^6
Ivory Coast	24 110.3	887 724.
Sudan	43 856.	1.63746×10^6
India	2.25189×10^6	8.98084×10^7
Morocco	65 678.6	2.33823×10^6
Eswatini	2065.37	94 628.4
Gambia	2429.38	69 760.7
Central African Republic	8405.99	244 859.
Indonesia	592 927.	2.24766×10^7
Cameroon	36 564.6	1.11578×10^6
Liberia	8106.76	227 678.
Tanzania	81 309.8	2.42555×10^6
Kenya	74 317.2	2.41546×10^6

Cambodia	31 158.4	888 669.
Sri Lanka	70 784.2	3.18789×10^6
Congo – Brazzaville	8526.96	246 897.
Zambia	29 447.8	917 387.
Congo – Kinshasa	140 748.	3.12933×10^6
Senegal	30 169.8	608 960.
Honduras	20 260.1	594 634.
Togo	15 313.6	342 603.
Bolivia	31 464.9	1.0349×10^6
Peru	101 351.	3.4095×10^6
Botswana	5198.8	122 016.
Syria	52 377.8	1.34569×10^6
Bulgaria	54 398.7	2.28878×10^6
Philippines	296 386.	9.31573×10^6
Belize	925.783	33 641.9
Guatemala	47 361.1	999 223.
Haiti	37 503.6	681 078.
Mongolia	11 576.5	347 787.
Tunisia	44 845.4	835 950.
Mauritius	7113.24	169 688.
Thailand	353 410.	7.39232×10^6
Niger	48 041.4	610 998.
Paraguay	26 575.2	580 037.
Jordan	21 588.4	358 293.
Fiji	5770.16	122 835.
Iraq	134 160.	1.69466×10^6
Argentina	277 379.	6.27671×10^6
Dominican Republic	71 624.	987 882.
Romania	279 344.	5.63083×10^6
Brazil	1.65424×10^6	1.78808×10^7
Panama	27 566.2	416 059.
Ghana	161 152.	1.67069×10^6
Albania	39 936.	504 643.
South Korea	570 145.	9.1123×10^6

Nicaragua	49 096.	502 169.
Costa Rica	36 409.8	466 409.
Namibia	16 345.1	185 637.
Mauritania	29 291.7	195 841.
Poland	691 033.	8.87071×10^6
Chile	222 999.	2.7243×10^6
Ecuador	157 048.	1.57675×10^6
Malaysia	294 342.	2.4573×10^6
Jamaica	46 413.9	462 320.
Iran	834 214.	4.68054×10^6
Angola	180 757.	868 413.
Malta	7587.92	72 038.3
Colombia	625 403.	4.81196×10^6
Barbados	5990.14	54 364.4
Taiwan	429 108.	3.4152×10^6
South Africa	736 824.	5.32956×10^6
Mexico	1.80238×10^6	1.30068×10^7
Trinidad and Tobago	31 091.6	250 555.
Gabon	21 116.7	99 604.8
Turkey	1.30078×10^6	6.46007×10^6
Uruguay	89 752.8	625 353.
Nigeria	2 279 523	8.76311×10^6
Hong Kong	152 682.	1.09687×10^6
Macao	7763.5	43 529.4
Hungary	370 264.	2.84892×10^6
Singapore	90 172.	398 148.
Portugal	367 140.	1.60443×10^6
Venezuela	673 383.	2.70873×10^6
Algeria	882 874.	2.40752×10^6
Ireland	158 013.	875 851.
Bahrain	17 677.7	62 273.5
New Zealand	188 041.	1.00861×10^6
United Kingdom	3.38196×10^6	1.67071×10^7

Austria	476 009.	2.12511×10^6
Israel	250 977.	1.08014×10^6
Spain	2.76914×10^6	8.3166×10^6
Saudi Arabia	825 502.	1.66064×10^6
Kuwait	116 840.	240 700.
Italy	4.85355×10^6	1.29119×10^7
Greece	871 054.	2.19958×10^6
Japan	10 868 648	3.54063×10^7
Canada	2 271 029	7.53408×10^6
Germany	7.8564×10^6	2.48823×10^7
Qatar	22 624.5	39 829.9
Belgium	1.00199×10^6	2.53485×10^6
Denmark	525 133.	1.50796×10^6
Netherlands	1 451 599	4.02862×10^6
Sweden	855 927	2.44125×10^6
Cyprus	52 916.8	107 905.
Australia	1.53587×10^6	4.68784×10^6
Finland	538 971.	1.29952×10^6
France	6.40826×10^6	1.42675×10^7
Norway	477 695.	1.22857×10^6
United States	28 983 498	7.67759×10^7
Iceland	33 001.8	55 408.4
Luxembourg	61 109.6	94 614.5
Switzerland	1.12533×10^6	2.09043×10^6
Brunei	39 051.9	40 988.5
United Arab Emirates	706 334.	164 904.

1990

1990 - Model Setup

1990 - Basic Model

In[•]:=
```
Clear[Activities, i, Wage, Profit, u];
(Activities = Table[Table[{0.0, 0.0, 0.0, 0.0, 0.0, 0.0}, {N}], {T}]);
Wage = Table[0.0, {T}];
Profit = Table[0.0, {T}];

(* First Stage t=1 simulation *)
(* Setting w1 and π1 *)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]], Wage[[1]] = b, Wage[[1]] = RandomReal[{b, (1 - A (1 + 0))/L}]],
  Wage[[1]] = (1 - A (1 + 0))/L];
Profit[[1]] = ((p - p0 A - Wage[[1]] L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^ν, y_t^ν, z_t^ν, δ_t^ν) for all ν and t=1*)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]],
   CapitalConstrained[1, A, L, l, 1, Profit[[1]], b], KnifeEdge[1, A, L, l, 1, Profit[[1]], b]],
  LabourConstrained[1, A, L, l, 1, Profit[[1]], b]
 ];

(* Simulation for remaining T-1 time steps *)
For[t = 2, t ≤ T, t++,

 (* Updating wt *)
```

```
If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧], If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
   Wage⟦t⟧ = b, Wage⟦t⟧ = RandomReal[{b, ((1 - A (1 + r))/L /. r → 0)}]], Wage⟦t⟧ = ((1 - A (1 + r))/L /. r → 0)];

(* Updating r_t *)
Profit⟦t⟧ = ((p - p0 A - Wage⟦t⟧ L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^v, y_t^v, z_t^v, δ_t^v) for all v *)
If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧],
 If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
  CapitalConstrained[1, A, L, l, t, Profit⟦t⟧, b], KnifeEdge[1, A, L, l, t, Profit⟦t⟧, b]],
 LabourConstrained[1, A, L, l, t, Profit⟦t⟧, b]
];

]
```

Exploitation and Class Over Time

The chart below captures the number of agents who are exploiters, exploited, or neither over the course of the simulation according to Definition 2, which defines exploitation in relation to Λ_t^v and $v_t\, c_t^v$, where v_t is just the embodied labour value and $c_t^v = \pi_t\, W_{t-1}^v + (w_t - b_t)\, l^v + b_t\, \Lambda_t^v$. An agent v is exploited during t if and only if $\Lambda_t^v > v_t\, c_t^v$, an agent is an exploiter if and only if $\Lambda_t^v < v_t\, c_t^v$, and an agent is neither exploited nor an exploiter if and only if $\Lambda_t^v = v_t\, c_t^v$.

```
In[•]:= Clear[IndivExploitation, ConsumptionBundle];
ConsumptionBundle = Table[0.0, {i, T}, {j, N}];
IndivExploitation = Table[0.0, {i, T}, {j, N}];

For[i = 1, i ≤ N, i++,
 ConsumptionBundle⟦1, i⟧ = EmbodiedValues⟦1, 2⟧ (Profit⟦1⟧ × AgentSet⟦i, 1⟧ + (Wage⟦1⟧ - b) l⟦i⟧ + b Activities⟦1, i, 6⟧);
```

```
 IndivExploitation⟦1, i⟧ = Activities⟦1, i, 6⟧ - ConsumptionBundle⟦1, i⟧;

]

For[t = 2, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ConsumptionBundle⟦t, i⟧ =
   EmbodiedValues⟦1, 2⟧ (Profit⟦t⟧ × Activities⟦t - 1, i, 5⟧ + (Wage⟦t⟧ - b) l⟦i⟧ + b Activities⟦t, i, 6⟧);
  IndivExploitation⟦t, i⟧ = Activities⟦t, i, 6⟧ - ConsumptionBundle⟦t, i⟧;
 ]
]

Clear[Exploiters, Exploited, NonExploit];
Exploiters = Table[0.0, {T}];
Exploited = Table[0.0, {T}];
NonExploit = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  If[IndivExploitation⟦t, i⟧ < 0, Exploiters⟦t⟧ ++, If[IndivExploitation⟦t, i⟧ == 0, NonExploit⟦t⟧ ++, Exploited⟦t⟧ ++]];
 ]
]

ExploitationLegend =
  Grid[{{Row[{Graphics[{Thick, Blue, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploiters", 20], FontFamily → "Times"]}], Spacer[10],
      Row[{Graphics[{Thick, Red, Dotted, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Neither Exploiter or Exploited", 20], FontFamily → "Times"]}]},
     {Row[{Graphics[{Thick, Black, Dashed, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploited", 20], FontFamily → "Times"]}], Spacer[10],
     }}, Frame → True, Alignment → Left];
(* ExploitationPlot=Labeled[
```

```
  ListLinePlot[{Exploiters,Exploited,NonExploit},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black},{Thick,Dotted,Red}},
   Frame→True,FrameLabel→{"t","Total ν in Group"},LabelStyle→20,
   PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],ExploitationLegend] *)
```

In[∘]:=
```
Export["./ExploitationPlot.eps", ExploitationPlot, "EPS"]
```

Out[∘]=
```
./ExploitationPlot.eps
```

The chart below captures the class composition of the simulation over time according to Corollary 1 of Theorem 3 (class is defined using labour endowments l^v in relation to means of production).

In[∘]:=
```
Clear[Class1, Class2, Class3, Class4, j, k];
Class1 = Table[0.0, {T}];
Class2 = Table[0.0, {T}];
Class3 = Table[0.0, {T}];
Class4 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ == Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class2⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class3⟦j⟧++, 0];
```

```
 ]
]

Clear[k];
For[k = 1, k ≤ N, k++,
 If[AgentSet⟦k, 1⟧ == 0 && Profit⟦1⟧ > 0, Class4⟦1⟧ ++, 0];
]

Clear[j, k];
For[j = 2, j ≤ T, j ++,
 For[k = 1, k ≤ N, k++,
  If[Activities⟦j - 1, k, 5⟧ == 0 && Profit⟦j⟧ > 0, Class4⟦j⟧ ++, 0];
 ]
]


(* ClassLegend=
 Column[{
   Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^1 = {∑v ∈ (+,0,+)\(+,0,0); A_t y_t^v < z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,DotDashed,Green,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^2 = {∑v ∈ (+,0,0); A_t y_t^v = z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Purple,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^3 = {∑v ∈ (+,+,0)\(+,0,0); A_t y_t^v > z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Black,Dotted,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^4 = {∑v ∈ (0,+,0); W_{t-1}^v = 0}",20],FontFamily→"Times"]}]
  },Frame→True,Alignment→Left];
ClassPlotCorollary1=Labeled[
  ListLinePlot[{Class1,Class2,Class3,Class4},PlotStyle→{{Thick,Blue},{Thick,DotDashed,Green},
     {Thick,Dashed,Purple},{Thick, Dotted,Black},{Thick,DotDashed,Orange},{Thick,Dashed,Green}},
```

```
   PlotRange→{{1,T},{-10,N+10}},Frame→True,FrameLabel→{"t","Total ν in Classes"},
   LabelStyle→20,ImageSize→{500,350},AxesOrigin→{1,-10}],ClassLegend] *)
```

In[∘]:=
```
(* Export["./ClassPlotCorollary1.eps",ClassPlotCorollary1,"EPS"] *)
```

The chart below captures the intersection of class and exploitation over the simulation. If an agent is in C^1 according to Corollary 1 of Theorem 3 and is also an exploiter according to Definition 2, then the agent is in the group "$C^1 \wedge$ Exploiter". If an agent is in $C^3 \cup C^4$ and is exploited they are in group "$(C^3 \cup C^4) \wedge$ Exploited".

In[∘]:=
```
Clear[CECP1, CECP2, CECP3, j, k];
CECP1 = Table[0.0, {T}];
CECP2 = Table[0.0, {T}];
CECP3a = Table[0.0, {T}];
CECP3b = Table[0.0, {T}];
CECP3 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ < 0, CECP1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
  If[If[j == 1, AgentSet⟦k, 1⟧, Activities⟦j - 1, k, 5⟧] == 0 && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
 ]
]


(* CECPLegend=
 Grid[{{Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],Spacer[5],
```

```
      Style[Style["∑v ∈ C_t^1 ∧ Exploiter",20],FontFamily→"Times"]}],Spacer[10],
    Row[{Graphics[{Thick,Black,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
      Spacer[5],Style[Style["∑v ∈ (C_t^3 ⋃ C_t^4) ∧ Exploited",20],FontFamily→"Times"]}],
   }},Frame→True,Alignment→Left];
CECPPlotCorollary1=
 Labeled[ListLinePlot[{CECP1,CECP3},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black}},Frame→True,FrameLabel→
    {"t","Total v in Group"},LabelStyle→20,PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],CECPLegend] *)
```

In[∘]:=
```
(* Export["./CECPPlotCorollary1.eps",CECPPlotCorollary1,"EPS"] *)
```

The charts below display the distribution of an index of the intensity of exploitation across the agents over the simulation. The index itself is calculated as $e_t^v = \Lambda_t^v / v_t c_t^v$ and the charts that follow explore some possibilities for visualizing the intensity of exploitation over time.

In[∘]:=
```
Clear[ExploitationIndex];
ExploitationIndex = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndex[[t, i]] = Activities[[t, i, 6]] / (EmbodiedValues[[1, 2]] (ConsumptionBundle[[t, i]] / EmbodiedValues[[1, 2]])) /. p → 1
 ]
]
```

The chart below uses *Mathematica's* `ArrayPlot` function to display the exploitation index of the N agents over *T*. There is a fixed distribution of uneven exploitation intensity while the simulation is capital constrained, however, this pattern disappears once the simulation is labour constrained, at which point exploitation intensity is the same for all $v \in \mathcal{N}$.

In[∘]:=
```
ExploitationIndex[[1, 1]]
```

Out[∘]=
```
1.13316
```

In[∘]:=
```
ExploitationIndex[[1]]
```

Out[∘]=
```
{1.13316, 1.13198, 1.13065, 1.13036, 1.13099, 1.12969, 1.12634, 1.12918, 1.12715, 1.12748, 1.12508,
 1.1236, 1.11857, 1.12507, 1.12513, 1.1211, 1.11861, 1.12203, 1.12091, 1.11532, 1.1126, 1.11612, 1.11203,
 1.11154, 1.11382, 1.11688, 1.11141, 1.11598, 1.10322, 1.1114, 1.10735, 1.11326, 1.09643, 1.10983, 1.11378,
 1.09773, 1.1032, 1.10586, 1.10828, 1.11423, 1.10469, 1.08137, 1.09098, 1.1121, 1.09921, 1.08592, 1.08975,
 1.09345, 1.07452, 1.09771, 1.09902, 1.06301, 1.09399, 1.10158, 1.0912, 1.07832, 1.07768, 1.07791, 1.07383,
 1.07403, 1.05287, 1.07217, 1.08016, 1.03836, 1.06997, 1.08466, 1.07788, 1.06802, 1.05513, 1.06854,
 1.08275, 1.08068, 1.03962, 1.00166, 1.01392, 1.07248, 1.06674, 1.01228, 1.04865, 1.05827, 1.05897,
 1.02238, 1.05223, 1.04466, 1.01678, 1.04642, 1.00889, 0.994909, 0.988159, 1.00821, 1.0246, 1.02684,
 1.00853, 0.997368, 1.00262, 0.954802, 1.00948, 0.999532, 0.99213, 0.997223, 0.977076, 0.965824, 0.989184,
 0.927051, 0.954434, 0.86348, 0.903926, 0.906344, 0.948333, 0.936079, 0.876487, 0.920528, 0.857736,
 0.9364, 0.935281, 0.831683, 0.931886, 0.875363, 0.913938, 0.793718, 0.890546, 0.855837, 0.823949,
 0.857546, 0.830376, 0.859546, 0.845913, 0.830872, 0.850301, 0.79979, 0.8624, 0.821605, 0.753927, 0.845906,
 0.826835, 0.802303, 0.8146, 0.804751, 0.817453, 0.713064, 0.656668, 0.733999, 0.661216, 0.299658}
```

In[∘]:=
```
CountryList
```

Out[∘]=
```
{Myanmar, Rwanda, Yemen, Ethiopia, Sierra Leone, Mali, Mozambique, Vietnam, Uganda, El Salvador, Madagascar, Malawi,
 Burkina Faso, Egypt, Syria, Nepal, Burundi, Bangladesh, Laos, Pakistan, Sudan, India, Central African Republic,
 Benin, Cambodia, Zimbabwe, Liberia, Lesotho, Gambia, Kenya, Togo, China, Senegal, Zambia, Bolivia, Ivory Coast,
 Cameroon, Maldives, Indonesia, Sri Lanka, Honduras, Niger, Tanzania, Belize, Congo - Brazzaville, Congo - Kinshasa,
 Guatemala, Eswatini, Haiti, Philippines, Mongolia, Nigeria, Fiji, Bulgaria, Peru, Ghana, Jordan, Botswana,
 Dominican Republic, Namibia, Iraq, Mauritius, Panama, Morocco, Paraguay, Kyrgyzstan, Jamaica, Thailand,
 Nicaragua, Costa Rica, Armenia, Tajikistan, Tunisia, Angola, Mauritania, Romania, Albania, Iran, Ecuador,
 Argentina, Barbados, Brazil, Moldova, Trinidad and Tobago, South Africa, Chile, Colombia, Gabon, Turkey,
 Malaysia, Lithuania, Poland, Uruguay, Mexico, Kazakhstan, Venezuela, South Korea, Croatia, Malta, Hungary,
 Serbia, Taiwan, Estonia, Macao, Latvia, Algeria, Qatar, Bahrain, Ukraine, Ireland, Singapore, Hong Kong,
 Saudi Arabia, Slovakia, United Kingdom, Portugal, New Zealand, Spain, Israel, Kuwait, Slovenia, Russia, Greece,
 Czech Republic, Belgium, Japan, Netherlands, France, Denmark, Italy, Germany, Austria, Cyprus, Australia,
 Canada, Finland, Sweden, Norway, United States, Iceland, Brunei, Switzerland, Luxembourg, United Arab Emirates}
```

In[ ]:=
```
(ExploitTable = Table[{CountryList⟦i⟧, ExploitationIndex⟦1, i⟧, DataNoHeader⟦i, 5⟧}, {i, 1, N}]) // TableForm
Export["./ExploitTable1990.csv", ExploitTable, "CSV"]
```

Out[ ]//TableForm=

Myanmar	1.13316	442.399
Rwanda	1.13198	554.245
Yemen	1.13065	593.981
Ethiopia	1.13036	645.014
Sierra Leone	1.13099	664.109
Mali	1.12969	735.859
Mozambique	1.12634	1126.07
Vietnam	1.12918	1221.43
Uganda	1.12715	1340.61
El Salvador	1.12748	1456.64
Madagascar	1.12508	1543.64
Malawi	1.1236	1748.05
Burkina Faso	1.11857	1853.38
Egypt	1.12507	1913.48
Syria	1.12513	1981.89
Nepal	1.1211	2017.59
Burundi	1.11861	2036.65
Bangladesh	1.12203	2114.38
Laos	1.12091	2360.08
Pakistan	1.11532	2898.85
Sudan	1.1126	2954.17
India	1.11612	3051.08
Central African Republic	1.11203	3103.09
Benin	1.11154	3139.59
Cambodia	1.11382	3281.8
Zimbabwe	1.11688	3558.62
Liberia	1.11141	3600.4
Lesotho	1.11598	3775.02
Gambia	1.10322	4011.6
Kenya	1.1114	4335.04
Togo	1.10735	4422.41
China	1.11326	4592.12
Senegal	1.09643	4883.59

Zambia	1.10983	4952.17
Bolivia	1.11378	4967.68
Ivory Coast	1.09773	5082.51
Cameroon	1.1032	5171.9
Maldives	1.10586	5353.17
Indonesia	1.10828	5433.98
Sri Lanka	1.11423	5630.63
Honduras	1.10469	5733.35
Niger	1.08137	6081.22
Tanzania	1.09098	6173.66
Belize	1.1121	6415.57
Congo – Brazzaville	1.09921	6797.93
Congo – Kinshasa	1.08592	7102.49
Guatemala	1.08975	7284.19
Eswatini	1.09345	7857.7
Haiti	1.07452	8782.2
Philippines	1.09771	8845.94
Mongolia	1.09902	9538.06
Nigeria	1.06301	9544.29
Fiji	1.09399	9627.47
Bulgaria	1.10158	9808.68
Peru	1.0912	10 323.9
Ghana	1.07832	11 412.7
Jordan	1.07768	11 948.8
Botswana	1.07791	12 222.8
Dominican Republic	1.07383	12 632.7
Namibia	1.07403	13 182.3
Iraq	1.05287	13 751.1
Mauritius	1.07217	13 823.7
Panama	1.08016	13 967.1
Morocco	1.03836	14 172.4
Paraguay	1.06997	14 295.9
Kyrgyzstan	1.08466	14 342.6
Jamaica	1.07788	14 487.2
Thailand	1.06802	14 806.2
Nicaragua	1.05513	15 019.5

Costa Rica	1.06854	16 156.3
Armenia	1.08275	16 516.9
Tajikistan	1.08068	16 599.1
Tunisia	1.03962	16 672.
Angola	1.00166	17 314.6
Mauritania	1.01392	18 429.
Romania	1.07248	18 535.
Albania	1.06674	18 580.4
Iran	1.01228	19 594.7
Ecuador	1.04865	20 972.3
Argentina	1.05827	21 111.9
Barbados	1.05897	21 277.6
Brazil	1.02238	21 651.9
Moldova	1.05223	23 297.7
Trinidad and Tobago	1.04466	25 264.8
South Africa	1.01678	25 383.4
Chile	1.04642	25 621.5
Colombia	1.00889	27 455.3
Gabon	0.994909	28 396.
Turkey	0.988159	30 582.
Malaysia	1.00821	31 719.
Lithuania	1.0246	32 125.2
Poland	1.02684	32 966.8
Uruguay	1.00853	33 631.4
Mexico	0.997368	34 384.3
Kazakhstan	1.00262	38 772.1
Venezuela	0.954802	39 030.2
South Korea	1.00948	40 478.6
Croatia	0.999532	41 514.5
Malta	0.99213	42 198.9
Hungary	0.997223	44 254.3
Serbia	0.977076	46 968.8
Taiwan	0.965824	47 013.3
Estonia	0.989184	47 104.3
Macao	0.927051	50 595.4
Latvia	0.954434	52 806.7

Algeria	0.86348	54 089.6
Qatar	0.903926	57 133.8
Bahrain	0.906344	60 669.4
Ukraine	0.948333	62 032.6
Ireland	0.936079	66 057.
Singapore	0.876487	68 658.9
Hong Kong	0.920528	70 505.
Saudi Arabia	0.857736	73 440.
Slovakia	0.9364	74 579.4
United Kingdom	0.935281	77 764.3
Portugal	0.831683	80 402.2
New Zealand	0.931886	80 827.6
Spain	0.875363	81 983.8
Israel	0.913938	86 601.
Kuwait	0.793718	97 182.7
Slovenia	0.890546	97 417.
Russia	0.855837	100 989.
Greece	0.823949	108 521.
Czech Republic	0.857546	114 556.
Belgium	0.830376	114 979.
Japan	0.859546	115 787.
Netherlands	0.845913	116 486.
France	0.830872	116 513.
Denmark	0.850301	118 149.
Italy	0.79979	121 582.
Germany	0.8624	123 281.
Austria	0.821605	128 509.
Cyprus	0.753927	132 333.
Australia	0.845906	132 506.
Canada	0.826835	139 360.
Finland	0.802303	139 430.
Sweden	0.8146	140 109.
Norway	0.804751	149 871.
United States	0.817453	151 609.
Iceland	0.713064	177 365.
Brunei	0.656668	201 249.

```
Switzerland                0.733999   214 773.
Luxembourg                 0.661216   221 488.
United Arab Emirates       0.299658   636 313.
```

Out[•]= ./ExploitTable1990.csv

In[•]:=
```
(* ExploitationArray=ArrayPlot[Transpose[ExploitationIndex],
  FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./ExploitationArray.eps",ExploitationArray,"EPS"] *)
```

In[•]:=
```
ExploitationIndexDistrib = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndexDistrib[[t, i]] = {t, ExploitationIndex[[t, i]]}
 ]
]
```

Below the Gini coefficient for the exploitation intensity index is plotted over *T*. The pattern of the Gini coefficient is consistent with the previous charts depicting the exploitation intensity index.

In[•]:=
```
(* Gini=Table[0.0,{T}];
For[t=1,t≤T,t++,
 Gini[[t]]=N/(N-1) ((Σ_{i=1}^N Σ_{j=1}^N Abs[ExploitationIndex[[t,i]]-ExploitationIndex[[t,j]]])/(2 N^2 Mean[ExploitationIndex[[t]]]))
] *)
```

In[•]:=
```
Clear[SortExploitationIndex];
SortExploitationIndex = Table[Table[0.0, {N}], {t, 1, T}];
GiniTest = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortExploitationIndex[[t]] = Sort[ExploitationIndex[[t]]];
```

$$\text{GiniTest}[[t]] = \frac{N}{N-1}\sum_{i=1}^{N}((2\,i - N - 1)\ \text{SortExploitationIndex}[[t, i]]) \Big/ \left(N^2\ \text{Mean}[\text{SortExploitationIndex}[[t]]]\right)$$

```
]
```

In[•]:=
```
Chop[GiniTest]
```

Out[•]= {0.0664991}

In[•]:=
```
(* ExploitationGini=ListLinePlot[GiniTest,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: e_t^γ"}] *)
```

In[•]:=
```
(* Export["./ExploitationGini.eps",ExploitationGini,"EPS"] *)
```

Plotting Gini coefficient for wealth $W_{t-1} = p_{t-1}\,\omega_{t-1}$.

In[•]:=
```
Clear[SortWealth, WealthGini];
SortWealth = Table[Table[0.0, {N}], {t, 1, T}];
WealthGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortWealth[[t]] = If[t == 1, Sort[AgentSet[[All, 1]]], Sort[Activities[[t - 1, All, 5]]]];
```

$$\text{WealthGini}[[t]] = \frac{N}{N-1}\sum_{i=1}^{N}\frac{(2\,i - N - 1)\ \text{SortWealth}[[t, i]]}{N^2\ \text{Mean}[\text{SortWealth}[[t]]]}$$

```
]
(* WealthGiniPlot=
 ListLinePlot[WealthGini,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: W_{t-1}^γ"},LabelStyle→12,PlotRange→{0,1}] *)
```

In[•]:=
```
(* Export["./WealthGini.eps",WealthGiniPlot,"EPS"] *)
```

In[∘]:=
```
WealthGini
```
Out[∘]=
```
{0.840982}
```

The figure below plots the distribution of wealth for select *t*.

In[∘]:=
```
(* WealthDistribRow=GraphicsRow[{
    Histogram[AgentSet⟦All,1⟧,10,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 1",18],AxesLabel→{"ω_{t-1}^γ"}],
    Histogram[Activities⟦24,All,5⟧,20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 25",18],AxesLabel→{"ω_{t-1}^γ"}],
    Histogram[Activities⟦49,All,5⟧,20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 50",18],AxesLabel→{"ω_{t-1}^γ"}]
   },Spacings→{15,-20}
  ]
  Export["./WealthDistribRow.eps",WealthDistribRow,"EPS"]
*)
```

Income Figures

Figures on net and gross income.

In[•]:=
```
Clear[Income];
Income = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  Income⟦t, i⟧ = Profit⟦t⟧ × Activities⟦t, i, 5⟧ + Wage⟦t⟧ × Activities⟦t, i, 6⟧
 ]
]

IncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 IncomeGini⟦t⟧ = N/(N - 1) ((Sum[Sum[Abs[Income⟦t, i⟧ - Income⟦t, j⟧], {j, 1, N}], {i, 1, N}]) / (2 N^2 Mean[Income⟦t⟧]))
]
(* IncomeGiniPlot=ListLinePlot[IncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[IncomeGini]+0.1},AxesLabel→{"t","Gini: Net Income"},LabelStyle→14] *)
```

In[•]:=
```
(* Export["./NetIncomeGini.eps",IncomeGiniPlot,"EPS"] *)
```

In[•]:=
```
Clear[IncomeShares];
IncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  IncomeShares⟦t, i⟧ = Income⟦t, i⟧ / Total[Income⟦t⟧]
 ]
]
(* IncomeArray=ArrayPlot[Transpose[IncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},
  PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",ColorFunctionScaling→True,
  DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./NetIncomeArray.eps",IncomeArray,"EPS"] *)
```

In[•]:=
```
Clear[GrossIncome];
GrossIncome = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncome⟦t, i⟧ = (1 + Profit⟦t⟧) Activities⟦t, i, 5⟧ + Wage⟦t⟧ × Activities⟦t, i, 6⟧
 ]
]

GrossIncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
```

$$\text{GrossIncomeGini}[[t]] = \frac{N}{N-1}\left(\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\text{Abs}[\text{GrossIncome}[[t, i]] - \text{GrossIncome}[[t, j]]]\right) \Big/ \left(2\,N^2\,\text{Mean}[\text{GrossIncome}[[t]]]\right)\right)$$

```
]
(* GrossIncomeGiniPlot=ListLinePlot[GrossIncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[GrossIncomeGini]+0.1},AxesLabel→{"t","Gini: Income"},LabelStyle→14] *)
```

In[•]:=
```
(* Export["./GrossIncomeGini.eps",GrossIncomeGiniPlot,"EPS"] *)
```

In[•]:=
```
GrossIncomeGini
```

Out[•]=
```
{0.816194}
```

In[∘]:=
```
Clear[GrossIncomeShares];
GrossIncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncomeShares[[t, i]] = GrossIncome[[t, i]] / Total[GrossIncome[[t]]]
 ]
]
(* GrossIncomeArray=
 ArrayPlot[Transpose[GrossIncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[∘]:=
```
(* Export["./GrossIncomeArray.eps",GrossIncomeArray,"EPS"] *)
```

Updated Simulation Reporting

In[∘]:=
```
(ExploitedTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploitedTable1990.csv", ExploitedTable, "CSV"]
```

Out[∘]//TableForm=

Myanmar	&	1.13316	\\
Rwanda	&	1.13198	\\
Yemen	&	1.13065	\\
Ethiopia	&	1.13036	\\
Sierra Leone	&	1.13099	\\
Mali	&	1.12969	\\
Mozambique	&	1.12634	\\
Vietnam	&	1.12918	\\
Uganda	&	1.12715	\\
El Salvador	&	1.12748	\\
Madagascar	&	1.12508	\\
Malawi	&	1.1236	\\

```
Burkina Faso                &    1.11857    \\
Egypt                       &    1.12507    \\
Syria                       &    1.12513    \\
Nepal                       &    1.1211     \\
Burundi                     &    1.11861    \\
Bangladesh                  &    1.12203    \\
Laos                        &    1.12091    \\
Pakistan                    &    1.11532    \\
Sudan                       &    1.1126     \\
India                       &    1.11612    \\
Central African Republic    &    1.11203    \\
Benin                       &    1.11154    \\
Cambodia                    &    1.11382    \\
Zimbabwe                    &    1.11688    \\
Liberia                     &    1.11141    \\
Lesotho                     &    1.11598    \\
Gambia                      &    1.10322    \\
Kenya                       &    1.1114     \\
Togo                        &    1.10735    \\
China                       &    1.11326    \\
Senegal                     &    1.09643    \\
Zambia                      &    1.10983    \\
Bolivia                     &    1.11378    \\
Ivory Coast                 &    1.09773    \\
Cameroon                    &    1.1032     \\
Maldives                    &    1.10586    \\
Indonesia                   &    1.10828    \\
Sri Lanka                   &    1.11423    \\
Honduras                    &    1.10469    \\
Niger                       &    1.08137    \\
Tanzania                    &    1.09098    \\
Belize                      &    1.1121     \\
Congo - Brazzaville         &    1.09921    \\
Congo - Kinshasa            &    1.08592    \\
Guatemala                   &    1.08975    \\
Eswatini                    &    1.09345    \\
```

```
Haiti                    &   1.07452   \\
Philippines              &   1.09771   \\
Mongolia                 &   1.09902   \\
Nigeria                  &   1.06301   \\
Fiji                     &   1.09399   \\
Bulgaria                 &   1.10158   \\
Peru                     &   1.0912    \\
Ghana                    &   1.07832   \\
Jordan                   &   1.07768   \\
Botswana                 &   1.07791   \\
Dominican Republic       &   1.07383   \\
Namibia                  &   1.07403   \\
Iraq                     &   1.05287   \\
Mauritius                &   1.07217   \\
Panama                   &   1.08016   \\
Morocco                  &   1.03836   \\
Paraguay                 &   1.06997   \\
Kyrgyzstan               &   1.08466   \\
Jamaica                  &   1.07788   \\
Thailand                 &   1.06802   \\
Nicaragua                &   1.05513   \\
Costa Rica               &   1.06854   \\
Armenia                  &   1.08275   \\
Tajikistan               &   1.08068   \\
Tunisia                  &   1.03962   \\
Angola                   &   1.00166   \\
Mauritania               &   1.01392   \\
Romania                  &   1.07248   \\
Albania                  &   1.06674   \\
Iran                     &   1.01228   \\
Ecuador                  &   1.04865   \\
Argentina                &   1.05827   \\
Barbados                 &   1.05897   \\
Brazil                   &   1.02238   \\
Moldova                  &   1.05223   \\
Trinidad and Tobago      &   1.04466   \\
```

```
South Africa                &    1.01678    \\
Chile                       &    1.04642    \\
Colombia                    &    1.00889    \\
Malaysia                    &    1.00821    \\
Lithuania                   &    1.0246     \\
Poland                      &    1.02684    \\
Uruguay                     &    1.00853    \\
Kazakhstan                  &    1.00262    \\
South Korea                 &    1.00948    \\
```

Out[•]= ./ExploitedTable1990.csv

In[•]:=
```
(ExploiterTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploiterTable1990.csv", ExploiterTable, "CSV"]
```

Out[•]//TableForm=
```
Gabon                   &    0.994909    \\
Turkey                  &    0.988159    \\
Mexico                  &    0.997368    \\
Venezuela               &    0.954802    \\
Croatia                 &    0.999532    \\
Malta                   &    0.99213     \\
Hungary                 &    0.997223    \\
Serbia                  &    0.977076    \\
Taiwan                  &    0.965824    \\
Estonia                 &    0.989184    \\
Macao                   &    0.927051    \\
Latvia                  &    0.954434    \\
Algeria                 &    0.86348     \\
Qatar                   &    0.903926    \\
Bahrain                 &    0.906344    \\
Ukraine                 &    0.948333    \\
Ireland                 &    0.936079    \\
Singapore               &    0.876487    \\
Hong Kong               &    0.920528    \\
Saudi Arabia            &    0.857736    \\
Slovakia                &    0.9364      \\
```

```
United Kingdom          &    0.935281    \\
Portugal                &    0.831683    \\
New Zealand             &    0.931886    \\
Spain                   &    0.875363    \\
Israel                  &    0.913938    \\
Kuwait                  &    0.793718    \\
Slovenia                &    0.890546    \\
Russia                  &    0.855837    \\
Greece                  &    0.823949    \\
Czech Republic          &    0.857546    \\
Belgium                 &    0.830376    \\
Japan                   &    0.859546    \\
Netherlands             &    0.845913    \\
France                  &    0.830872    \\
Denmark                 &    0.850301    \\
Italy                   &    0.79979     \\
Germany                 &    0.8624      \\
Austria                 &    0.821605    \\
Cyprus                  &    0.753927    \\
Australia               &    0.845906    \\
Canada                  &    0.826835    \\
Finland                 &    0.802303    \\
Sweden                  &    0.8146      \\
Norway                  &    0.804751    \\
United States           &    0.817453    \\
Iceland                 &    0.713064    \\
Brunei                  &    0.656668    \\
Switzerland             &    0.733999    \\
Luxembourg              &    0.661216    \\
United Arab Emirates    &    0.299658    \\
```

Out[•]= ./ExploiterTable1990.csv

In[•]:=
```
(ExploitIncomeCSV = Table[{CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"}, {i, 1, N}]) // TableForm
Export["./ExploitIncomeTable1990.csv", ExploitIncomeCSV, "CSV"]
```

Out[•]//TableForm=
```
Myanmar                     &    1.13316     \\
Rwanda                      &    1.13198     \\
```

```
Yemen                      &   1.13065    \\
Ethiopia                   &   1.13036    \\
Sierra Leone               &   1.13099    \\
Mali                       &   1.12969    \\
Mozambique                 &   1.12634    \\
Vietnam                    &   1.12918    \\
Uganda                     &   1.12715    \\
El Salvador                &   1.12748    \\
Madagascar                 &   1.12508    \\
Malawi                     &   1.1236     \\
Burkina Faso               &   1.11857    \\
Egypt                      &   1.12507    \\
Syria                      &   1.12513    \\
Nepal                      &   1.1211     \\
Burundi                    &   1.11861    \\
Bangladesh                 &   1.12203    \\
Laos                       &   1.12091    \\
Pakistan                   &   1.11532    \\
Sudan                      &   1.1126     \\
India                      &   1.11612    \\
Central African Republic   &   1.11203    \\
Benin                      &   1.11154    \\
Cambodia                   &   1.11382    \\
Zimbabwe                   &   1.11688    \\
Liberia                    &   1.11141    \\
Lesotho                    &   1.11598    \\
Gambia                     &   1.10322    \\
Kenya                      &   1.1114     \\
Togo                       &   1.10735    \\
China                      &   1.11326    \\
Senegal                    &   1.09643    \\
Zambia                     &   1.10983    \\
Bolivia                    &   1.11378    \\
Ivory Coast                &   1.09773    \\
Cameroon                   &   1.1032     \\
Maldives                   &   1.10586    \\
```

```
Indonesia                  &    1.10828     \\
Sri Lanka                  &    1.11423     \\
Honduras                   &    1.10469     \\
Niger                      &    1.08137     \\
Tanzania                   &    1.09098     \\
Belize                     &    1.1121      \\
Congo - Brazzaville        &    1.09921     \\
Congo - Kinshasa           &    1.08592     \\
Guatemala                  &    1.08975     \\
Eswatini                   &    1.09345     \\
Haiti                      &    1.07452     \\
Philippines                &    1.09771     \\
Mongolia                   &    1.09902     \\
Nigeria                    &    1.06301     \\
Fiji                       &    1.09399     \\
Bulgaria                   &    1.10158     \\
Peru                       &    1.0912      \\
Ghana                      &    1.07832     \\
Jordan                     &    1.07768     \\
Botswana                   &    1.07791     \\
Dominican Republic         &    1.07383     \\
Namibia                    &    1.07403     \\
Iraq                       &    1.05287     \\
Mauritius                  &    1.07217     \\
Panama                     &    1.08016     \\
Morocco                    &    1.03836     \\
Paraguay                   &    1.06997     \\
Kyrgyzstan                 &    1.08466     \\
Jamaica                    &    1.07788     \\
Thailand                   &    1.06802     \\
Nicaragua                  &    1.05513     \\
Costa Rica                 &    1.06854     \\
Armenia                    &    1.08275     \\
Tajikistan                 &    1.08068     \\
Tunisia                    &    1.03962     \\
Angola                     &    1.00166     \\
```

```
Mauritania                &   1.01392    \\
Romania                   &   1.07248    \\
Albania                   &   1.06674    \\
Iran                      &   1.01228    \\
Ecuador                   &   1.04865    \\
Argentina                 &   1.05827    \\
Barbados                  &   1.05897    \\
Brazil                    &   1.02238    \\
Moldova                   &   1.05223    \\
Trinidad and Tobago       &   1.04466    \\
South Africa              &   1.01678    \\
Chile                     &   1.04642    \\
Colombia                  &   1.00889    \\
Gabon                     &   0.994909   \\
Turkey                    &   0.988159   \\
Malaysia                  &   1.00821    \\
Lithuania                 &   1.0246     \\
Poland                    &   1.02684    \\
Uruguay                   &   1.00853    \\
Mexico                    &   0.997368   \\
Kazakhstan                &   1.00262    \\
Venezuela                 &   0.954802   \\
South Korea               &   1.00948    \\
Croatia                   &   0.999532   \\
Malta                     &   0.99213    \\
Hungary                   &   0.997223   \\
Serbia                    &   0.977076   \\
Taiwan                    &   0.965824   \\
Estonia                   &   0.989184   \\
Macao                     &   0.927051   \\
Latvia                    &   0.954434   \\
Algeria                   &   0.86348    \\
Qatar                     &   0.903926   \\
Bahrain                   &   0.906344   \\
Ukraine                   &   0.948333   \\
Ireland                   &   0.936079   \\
```

```
Singapore                 &   0.876487   \\
Hong Kong                 &   0.920528   \\
Saudi Arabia              &   0.857736   \\
Slovakia                  &   0.9364     \\
United Kingdom            &   0.935281   \\
Portugal                  &   0.831683   \\
New Zealand               &   0.931886   \\
Spain                     &   0.875363   \\
Israel                    &   0.913938   \\
Kuwait                    &   0.793718   \\
Slovenia                  &   0.890546   \\
Russia                    &   0.855837   \\
Greece                    &   0.823949   \\
Czech Republic            &   0.857546   \\
Belgium                   &   0.830376   \\
Japan                     &   0.859546   \\
Netherlands               &   0.845913   \\
France                    &   0.830872   \\
Denmark                   &   0.850301   \\
Italy                     &   0.79979    \\
Germany                   &   0.8624     \\
Austria                   &   0.821605   \\
Cyprus                    &   0.753927   \\
Australia                 &   0.845906   \\
Canada                    &   0.826835   \\
Finland                   &   0.802303   \\
Sweden                    &   0.8146     \\
Norway                    &   0.804751   \\
United States             &   0.817453   \\
Iceland                   &   0.713064   \\
Brunei                    &   0.656668   \\
Switzerland               &   0.733999   \\
Luxembourg                &   0.661216   \\
United Arab Emirates      &   0.299658   \\
```

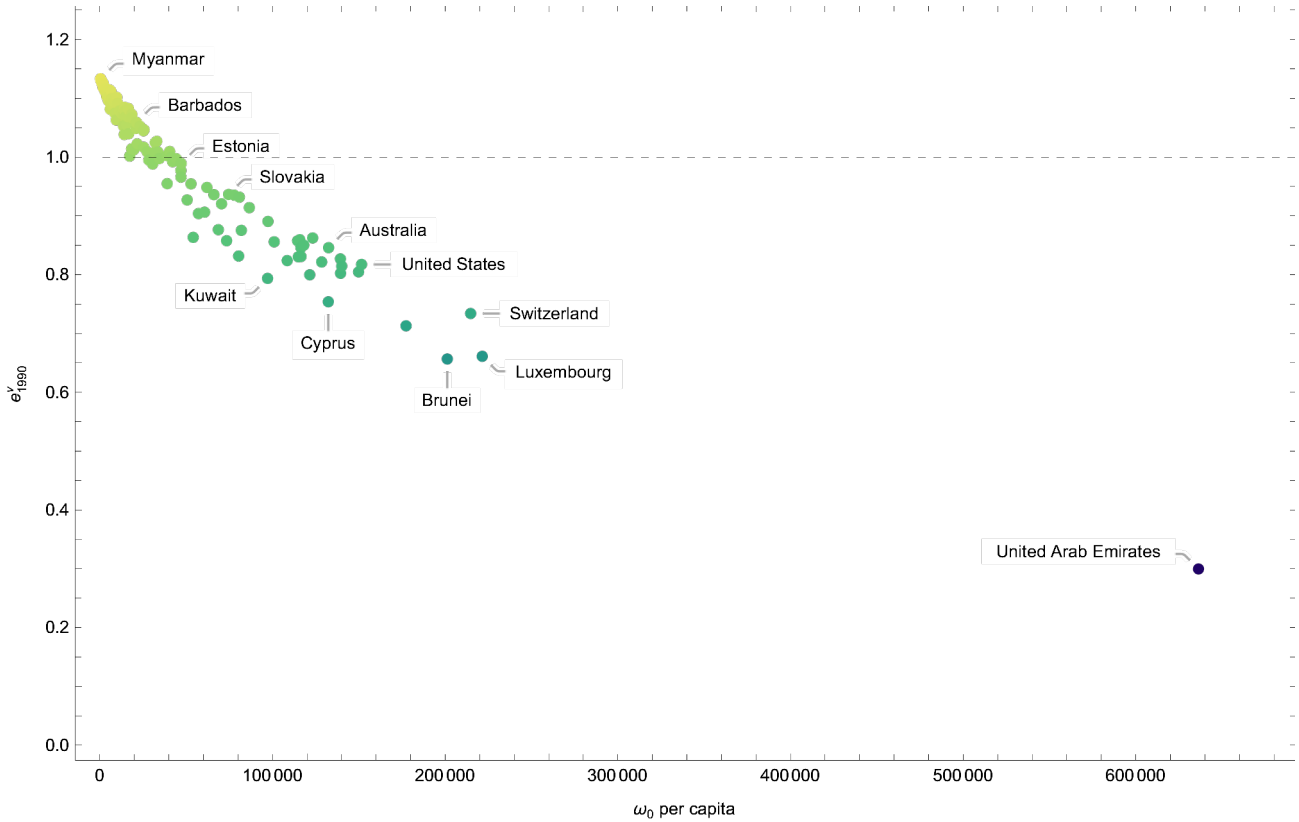
Out[•]= ./ExploitIncomeTable1990.csv

In[∘]:=
```
DataNoHeader[[1]]
```

Out[∘]=
```
{Myanmar, 18 286.6, 1.3849, 41.3352, 442.399}
```

In[∘]:=
```
ExploitWealthPlot = ListPlot[Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, 1, N}] → CountryList,
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^γ_1990"}, ColorFunction → "BlueGreenYellow",
  Epilog → {Black, Dashed, Line[{{0, 1}, {2 000 000, 1}}]}]
Export["./ExploitWealthPlot1990.eps", ExploitWealthPlot, "EPS"]
```

Out[∘]=



Out[∘]=
```
./ExploitWealthPlot1990.eps
```

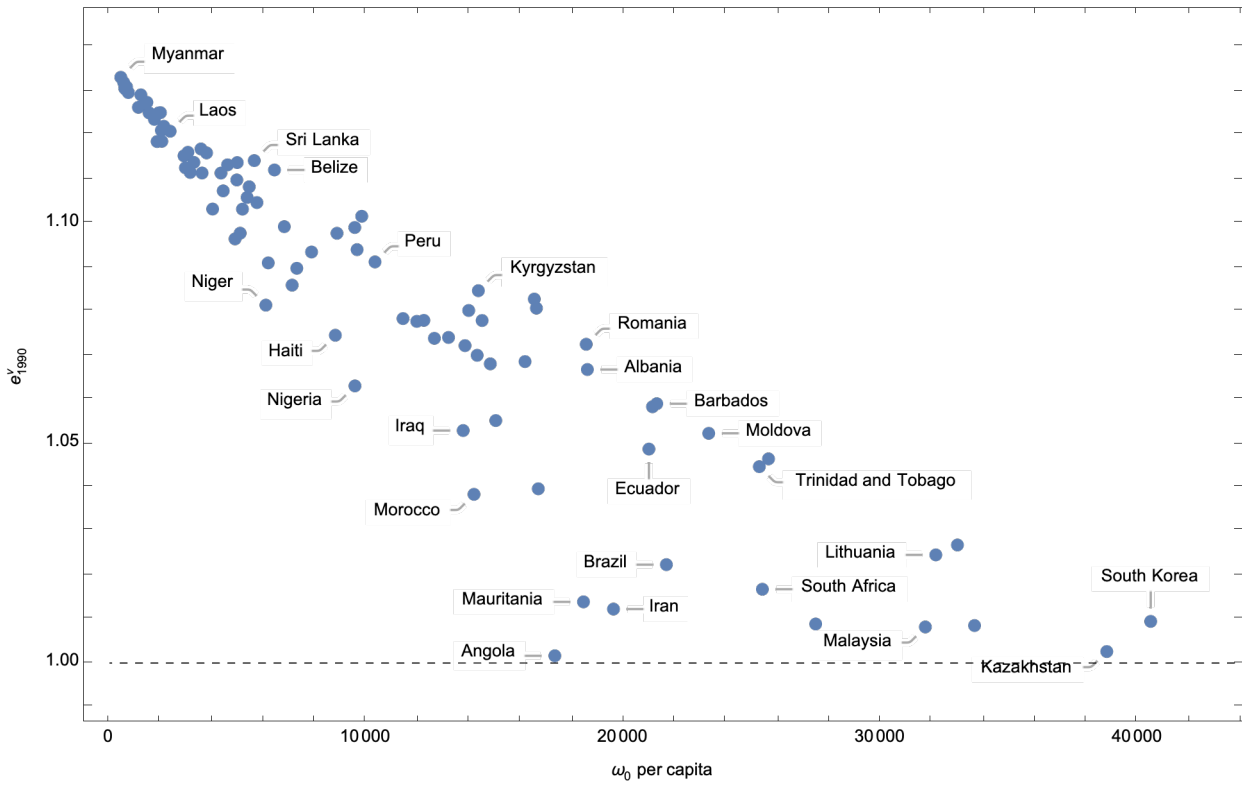
In[∘]:=
```
ExploitedCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1, i, 0.], {i, 1, N}], 0.]
ExploiterCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1, i, 0.], {i, 1, N}], 0.]
```

Out[∘]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64,
 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 90, 91, 92, 93, 95, 97}
```

Out[∘]=
```
{88, 89, 94, 96, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119,
 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144}
```

In[∘]:=
```
ExploitedCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploitedCountries}] → CountryList[[ExploitedCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^y_1990"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitedCountriesPlot1990.eps", ExploitedCountriesPlot, "EPS"]
```

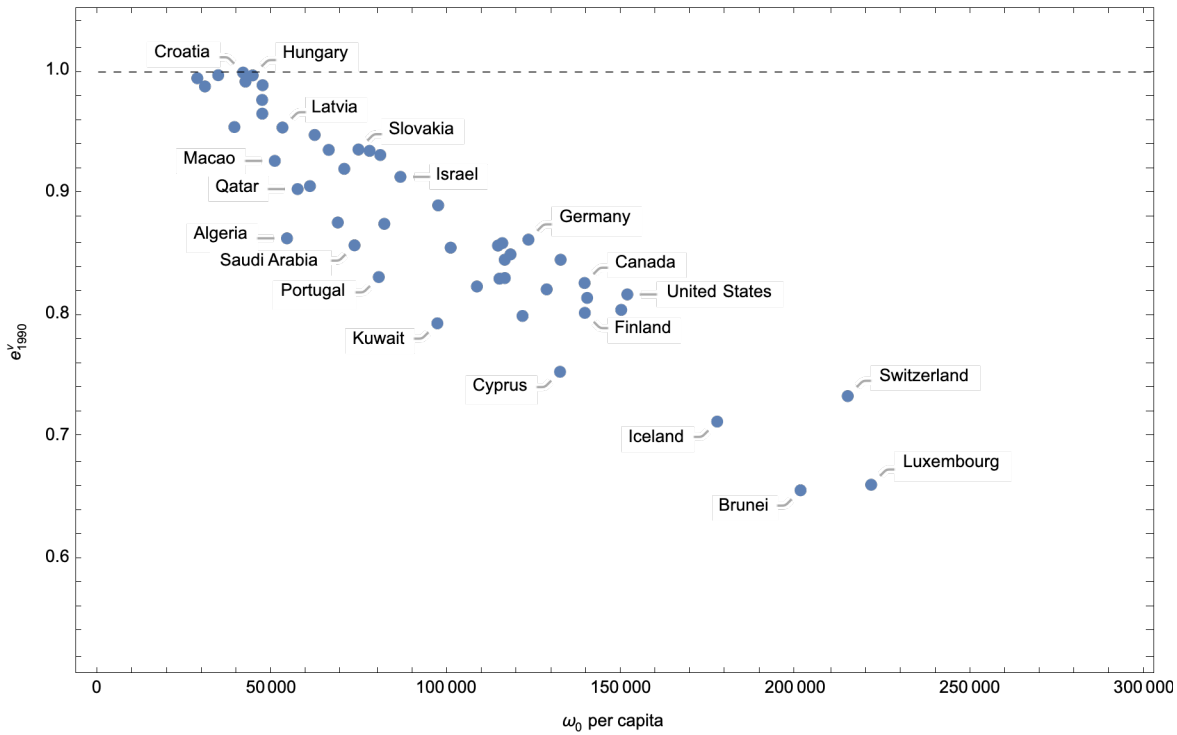
Out[∘]=



Out[∘]= ./ExploitedCountriesPlot1990.eps

In[∘]:=
```
ExploiterCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploiterCountries}] → CountryList[[ExploiterCountries]],
  LabelingFunction → Callout[Automatic, Automatic], Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_1990"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploiterCountriesPlot1990.eps", ExploiterCountriesPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploiterCountriesPlot1990.eps

In[∘]:=
```
(outlierTesting = Table[{CountryList[[i]], AgentSet[[i, 1]], AgentSet[[i, 2]]}, {i, 1, Length[CountryList]}]) // TableForm
Export["./outlierTesting1990.csv", outlierTesting, "CSV"];
```

Out[∘]//TableForm=

Myanmar	18 286.6	5.72451×10^6
Rwanda	4039.84	922 192.

Yemen	6955.52	1.21761×10^6
Ethiopia	30 888.4	5.14412×10^6
Sierra Leone	2868.82	533 625.
Mali	6217.93	930 925.
Mozambique	14 624.6	1.45198×10^6
Vietnam	83 043.9	1.15316×10^7
Uganda	23 265.6	2.51465×10^6
El Salvador	7676.64	861 345.
Madagascar	17 904.	1.57776×10^6
Malawi	16 439.5	1.27967×10^6
Burkina Faso	16 330.1	907 181.
Egypt	107 412.	9.46203×10^6
Syria	24 667.	2.18381×10^6
Nepal	38 143.6	2.47662×10^6
Burundi	11 077.4	616 946.
Bangladesh	218 145.	1.5092×10^7
Laos	10 050.4	644 226.
Pakistan	312 055.	1.46186×10^7
Sudan	59 519.5	2.46304×10^6
India	2.66444×10^6	1.29848×10^8
Central African Republic	8709.43	351 820.
Benin	15 630.4	618 628.
Cambodia	29 456.1	1.28647×10^6
Zimbabwe	37 125.	1.88075×10^6
Liberia	7474.07	294 259.
Lesotho	6431.5	311 266.
Gambia	3833.49	112 789.
Kenya	102 847.	4.04671×10^6
Togo	16 691.5	563 163.
China	5.40439×10^6	2.3021×10^8
Senegal	36 755.4	892 092.
Zambia	39 799.6	1.47146×10^6
Bolivia	34 102.2	1.48664×10^6

Ivory Coast	60 608.5	1.52209×10^6
Cameroon	60 925.5	1.79152×10^6
Maldives	1194.83	38 290.
Indonesia	989 807.	3.45231×10^7
Sri Lanka	97 555.2	4.34081×10^6
Honduras	28 410.5	875 899.
Niger	48 811.5	848 412.
Tanzania	151 400.	3.21701×10^6
Belize	1203.56	48 762.9
Congo – Brazzaville	16 020.7	419 021.
Congo – Kinshasa	245 831.	4.67747×10^6
Guatemala	67 479.3	1.39448×10^6
Eswatini	6462.17	145 557.
Haiti	61 808.2	949 342.
Philippines	547 521.	1.37432×10^7
Mongolia	20 832.1	541 984.
Nigeria	908 735.	1.16407×10^7
Fiji	7014.58	160 081.
Bulgaria	86 723.5	2.42743×10^6
Peru	227 863.	4.8663×10^6
Ghana	168 603.	2.76866×10^6
Jordan	42 608.1	691 642.
Botswana	15 728.3	256 356.
Dominican Republic	90 115.7	1.36778×10^6
Namibia	18 888.9	287 669.
Iraq	239 532	2.66983×10^6
Mauritius	14 596.4	215 488.
Panama	34 511.4	586 246.
Morocco	351 583.	3.2927×10^6
Paraguay	60 377.2	860 053.
Kyrgyzstan	62 718.9	1.16297×10^6
Jamaica	35 057.5	571 096.
Thailand	837 415.	1.15684×10^7
Nicaragua	62 682.4	719 703.

Costa Rica	50 398.	701 865.
Armenia	58 440.3	1.04327×10^6
Tajikistan	87 706.4	1.50472×10^6
Tunisia	137 419.	1.30536×10^6
Angola	205 150.	1.34847×10^6
Mauritania	37 490.1	274 407.
Romania	435 373.	6.46023×10^6
Albania	61 056.9	826 848.
Iran	1.10448×10^6	7.96454×10^6
Ecuador	214 566.	2.26758×10^6
Argentina	688 642.	8.24927×10^6
Barbados	5551.32	67 140.
Brazil	3.2262×10^6	2.55794×10^7
Moldova	101 709.	1.12436×10^6
Trinidad and Tobago	30 850.8	310 636.
South Africa	934 120.	7.0208×10^6
Chile	340 116.	3.49786×10^6
Colombia	908 843.	6.35802×10^6
Gabon	26 962.	167 625.
Turkey	1.64904×10^6	9.71887×10^6
Malaysia	571 888	3.97701×10^6
Lithuania	118 735.	962 143.
Poland	1.25142×10^6	1.03715×10^7
Uruguay	104 580.	729 326.
Mexico	2.88633×10^6	1.83072×10^7
Kazakhstan	635 239	4.20952×10^6
Venezuela	766 267.	3.56196×10^6
South Korea	1.73728×10^6	1.22172×10^7
Croatia	198 290.	1.28036×10^6
Malta	15 276.	92 882.
Hungary	459 231.	2.90932×10^6
Serbia	447 035.	2.42387×10^6

Taiwan	953 378.	4.77264×10^6
Estonia	73 727.7	438 006.
Macao	17 394.7	68 099.9
Latvia	140 698.	652 458.
Algeria	1.39329×10^6	3.89706×10^6
Qatar	27 212.8	93 545.3
Bahrain	30 086.	104 789.
Ukraine	3.19239×10^6	1.42321×10^7
Ireland	231 920.	958 125.
Singapore	206 869.	616 731.
Hong Kong	403 846.	1.52248×10^6
Saudi Arabia	1.19221×10^6	3.24416×10^6
Slovakia	394 413.	1.63261×10^6
United Kingdom	4.44301×10^6	1.8267×10^7
Portugal	795 612.	1.91971×10^6
New Zealand	274 668.	1.10649×10^6
Spain	3.21397×10^6	9.52817×10^6
Israel	385 236.	1.3992×10^6
Kuwait	203 627.	416 944.
Slovenia	195 458.	625 917.
Russia	14 899 058	4.01788×10^7
Greece	1.10973×10^6	2.58708×10^6
Czech Republic	1.18461×10^6	3.22057×10^6
Belgium	1.15054×10^6	2.75989×10^6
Japan	14 416 034	3.95678×10^7
Netherlands	1.74326×10^6	4.48782×10^6
France	6 785 194	1.63124×10^7
Denmark	607 417	1.59595×10^6
Italy	6 936 038	1.4569×10^7
Germany	9 745 840	2.71179×10^7
Austria	992 590.	2.29021×10^6
Cyprus	76 673.8	133 610.
Australia	2.24738×10^6	5.78543×10^6

Canada	3.83814×10^6	9.06274×10^6
Finland	696 621.	1.47888×10^6
Sweden	1.20037×10^6	2.68622×10^6
Norway	636 545.	1.36546×10^6
United States	38 223 704	8.66058×10^7
Iceland	45 228.	67 345.5
Brunei	52 063.	62 998.6
Switzerland	1.42887×10^6	2.30403×10^6
Luxembourg	84 564.1	104 021.
United Arab Emirates	1.16344×10^6	368 313.

2000

2000 - Model Setup

2000 - Basic Model

In[ ]:=
```
Clear[Activities, i, Wage, Profit, u];
(Activities = Table[Table[{0.0, 0.0, 0.0, 0.0, 0.0, 0.0}, {N}], {T}]);
Wage = Table[0.0, {T}];
Profit = Table[0.0, {T}];

(* First Stage t=1 simulation *)
(* Setting w1 and π1 *)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]], Wage[[1]] = b, Wage[[1]] = RandomReal[{b, (1 - A (1 + 0))/L}]],
  Wage[[1]] = (1 - A (1 + 0))/L];
```

```
Profit⟦1⟧ = ((p - p0 A - Wage⟦1⟧ L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^ν,y_t^ν,z_t^ν,δ_t^ν) for all ν and t=1*)
If[Total[l] ≥ L A^-1 Total[AgentSet⟦All, 1⟧],
  If[Total[l] > L A^-1 Total[AgentSet⟦All, 1⟧],
   CapitalConstrained[1, A, L, l, 1, Profit⟦1⟧, b], KnifeEdge[1, A, L, l, 1, Profit⟦1⟧, b]],
  LabourConstrained[1, A, L, l, 1, Profit⟦1⟧, b]
 ];

(* Simulation for remaining T-1 time steps *)
For[t = 2, t ≤ T, t++,

 (* Updating w_t *)
 If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧], If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
   Wage⟦t⟧ = b, Wage⟦t⟧ = RandomReal[{b, ((1 - A (1 + r))/L /. r → 0)}]], Wage⟦t⟧ = ((1 - A (1 + r))/L /. r → 0)];

 (* Updating r_t *)
 Profit⟦t⟧ = ((p - p0 A - Wage⟦t⟧ L)/(p0 A)) /. {p → 1, p0 → 1};

 (* Checking whether the simulation is capital constrained, labour constrained,
 or on the knife-edge and calling the corresponding function to find optimal (x_t^ν,y_t^ν,z_t^ν,δ_t^ν) for all ν *)
 If[Total[l] ≥ L A^-1 Total[Activities⟦t - 1, All, 5⟧],
  If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
   CapitalConstrained[1, A, L, l, t, Profit⟦t⟧, b], KnifeEdge[1, A, L, l, t, Profit⟦t⟧, b]],
  LabourConstrained[1, A, L, l, t, Profit⟦t⟧, b]
 ];
```

]

Exploitation and Class Over Time

The chart below captures the number of agents who are exploiters, exploited, or neither over the course of the simulation according to Definition 2, which defines exploitation in relation to Λ_t^v and $v_t\, c_t^v$, where v_t is just the embodied labour value and $c_t^v = \pi_t\, W_{t-1}^v + (w_t - b_t)\; l^v + b_t\, \Lambda_t^v$. An agent v is exploited during t if and only if $\Lambda_t^v > v_t\, c_t^v$, an agent is an exploiter if and only if $\Lambda_t^v < v_t\, c_t^v$, and an agent is neither exploited nor an exploiter if and only if $\Lambda_t^v = v_t\, c_t^v$.

In[ ]:=
```
Clear[IndivExploitation, ConsumptionBundle];
ConsumptionBundle = Table[0.0, {i, T}, {j, N}];
IndivExploitation = Table[0.0, {i, T}, {j, N}];

For[i = 1, i ≤ N, i++,
 ConsumptionBundle⟦1, i⟧ = EmbodiedValues⟦1, 2⟧ (Profit⟦1⟧ × AgentSet⟦i, 1⟧ + (Wage⟦1⟧ - b) l⟦i⟧ + b Activities⟦1, i, 6⟧);
 IndivExploitation⟦1, i⟧ = Activities⟦1, i, 6⟧ - ConsumptionBundle⟦1, i⟧;

]

For[t = 2, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ConsumptionBundle⟦t, i⟧ =
   EmbodiedValues⟦1, 2⟧ (Profit⟦t⟧ × Activities⟦t - 1, i, 5⟧ + (Wage⟦t⟧ - b) l⟦i⟧ + b Activities⟦t, i, 6⟧);
  IndivExploitation⟦t, i⟧ = Activities⟦t, i, 6⟧ - ConsumptionBundle⟦t, i⟧;
 ]
]

Clear[Exploiters, Exploited, NonExploit];
Exploiters = Table[0.0, {T}];
Exploited = Table[0.0, {T}];
NonExploit = Table[0.0, {T}];
```

```
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  If[IndivExploitation⟦t, i⟧ < 0, Exploiters⟦t⟧ ++, If[IndivExploitation⟦t, i⟧ == 0, NonExploit⟦t⟧ ++, Exploited⟦t⟧ ++]];
 ]
]

ExploitationLegend =
  Grid[{{Row[{Graphics[{Thick, Blue, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploiters", 20], FontFamily → "Times"]}], Spacer[10],
      Row[{Graphics[{Thick, Red, Dotted, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Neither Exploiter or Exploited", 20], FontFamily → "Times"]}]},
     {Row[{Graphics[{Thick, Black, Dashed, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploited", 20], FontFamily → "Times"]}], Spacer[10],
     }}, Frame → True, Alignment → Left];
(* ExploitationPlot=Labeled[
  ListLinePlot[{Exploiters,Exploited,NonExploit},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black},{Thick,Dotted,Red}},
   Frame→True,FrameLabel→{"t","Total ν in Group"},LabelStyle→20,
   PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],ExploitationLegend] *)
```

In[•]:=
```
Export["./ExploitationPlot.eps", ExploitationPlot, "EPS"]
```

Out[•]=
```
./ExploitationPlot.eps
```

The chart below captures the class composition of the simulation over time according to Corollary 1 of Theorem 3 (class is defined using labour endowments l^{ν} in relation to means of production).

In[•]:=
```
Clear[Class1, Class2, Class3, Class4, j, k];
Class1 = Table[0.0, {T}];
Class2 = Table[0.0, {T}];
Class3 = Table[0.0, {T}];
Class4 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
```

```
    If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class1⟦j⟧ ++, 0];
   ]
  ]

  Clear[j, k];
  For[j = 1, j ≤ T, j ++,
   For[k = 1, k ≤ N, k ++,
    If[A Activities⟦j, k, 2⟧ == Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class2⟦j⟧ ++, 0];
   ]
  ]

  Clear[j, k];
  For[j = 1, j ≤ T, j ++,
   For[k = 1, k ≤ N, k ++,
    If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class3⟦j⟧ ++, 0];
   ]
  ]

  Clear[k];
  For[k = 1, k ≤ N, k ++,
   If[AgentSet⟦k, 1⟧ == 0 && Profit⟦1⟧ > 0, Class4⟦1⟧ ++, 0];
  ]

  Clear[j, k];
  For[j = 2, j ≤ T, j ++,
   For[k = 1, k ≤ N, k ++,
    If[Activities⟦j - 1, k, 5⟧ == 0 && Profit⟦j⟧ > 0, Class4⟦j⟧ ++, 0];
   ]
  ]
```

```
(* ClassLegend=
 Column[{
   Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^1 = {∑v ∈ (+,0,+)\(+,0,0); A_t y_t^v < z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,DotDashed,Green,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^2 = {∑v ∈ (+,0,0); A_t y_t^v = z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Purple,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^3 = {∑v ∈ (+,+,0)\(+,0,0); A_t y_t^v > z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Black,Dotted,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^4 = {∑v ∈ (0,+,0); W_{t-1}^v = 0}",20],FontFamily→"Times"]}]
  },Frame→True,Alignment→Left];
ClassPlotCorollary1=Labeled[
  ListLinePlot[{Class1,Class2,Class3,Class4},PlotStyle→{{Thick,Blue},{Thick,DotDashed,Green},
     {Thick,Dashed,Purple},{Thick, Dotted,Black},{Thick,DotDashed,Orange},{Thick,Dashed,Green}},
   PlotRange→{{1,T},{-10,N+10}},Frame→True,FrameLabel→{"t","Total v in Classes"},
   LabelStyle→20,ImageSize→{500,350},AxesOrigin→{1,-10}],ClassLegend] *)
```

In[∘]:=
```
(* Export["./ClassPlotCorollary1.eps",ClassPlotCorollary1,"EPS"] *)
```

The chart below captures the intersection of class and exploitation over the simulation. If an agent is in C^1 according to Corollary 1 of Theorem 3 and is also an exploiter according to Definition 2, then the agent is in the group "$C^1 \wedge$ Exploiter". If an agent is in $C^3 \cup C^4$ and is exploited they are in group "$(C^3 \cup C^4) \wedge$ Exploited".

In[∘]:=
```
Clear[CECP1, CECP2, CECP3, j, k];
CECP1 = Table[0.0, {T}];
CECP2 = Table[0.0, {T}];
CECP3a = Table[0.0, {T}];
CECP3b = Table[0.0, {T}];
CECP3 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
```

```
    If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ < 0, CECP1⟦j⟧++, 0];
   ]
  ]

  Clear[j, k];
  For[j = 1, j ≤ T, j++,
   For[k = 1, k ≤ N, k++,
    If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
    If[If[j == 1, AgentSet⟦k, 1⟧, Activities⟦j - 1, k, 5⟧] == 0 && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
   ]
  ]


  (* CECPLegend=
   Grid[{{Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],Spacer[5],
        Style[Style["∑v ∈ C_t^1 ∧ Exploiter",20],FontFamily→"Times"]}],Spacer[10],
      Row[{Graphics[{Thick,Black,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
        Spacer[5],Style[Style["∑v ∈ (C_t^3 ⋃ C_t^4) ∧ Exploited",20],FontFamily→"Times"]}],
     }},Frame→True,Alignment→Left];
  CECPPlotCorollary1=
   Labeled[ListLinePlot[{CECP1,CECP3},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black}},Frame→True,FrameLabel→
      {"t","Total v in Group"},LabelStyle→20,PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],CECPLegend] *)
```

In[∘]:=
```
(* Export["./CECPPlotCorollary1.eps",CECPPlotCorollary1,"EPS"] *)
```

The charts below display the distribution of an index of the intensity of exploitation across the agents over the simulation. The index itself is calculated as $e_t^v = \Lambda_t^v / v_t c_t^v$ and the charts that follow explore some possibilities for visualizing the intensity of exploitation over time.

In[∘]:=
```
Clear[ExploitationIndex];
ExploitationIndex = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndex[[t, i]] = Activities[[t, i, 6]] / (EmbodiedValues[[1, 2]] ConsumptionBundle[[t, i]] / EmbodiedValues[[1, 2]]) /. p → 1
 ]
]
```

The chart below uses *Mathematica's* `ArrayPlot` function to display the exploitation index of the N agents over *T*. There is a fixed distribution of uneven exploitation intensity while the simulation is capital constrained, however, this pattern disappears once the simulation is labour constrained, at which point exploitation intensity is the same for all $v \in \mathcal{N}$.

In[∘]:=
```
ExploitationIndex[[1, 1]]
```

Out[∘]=
```
1.13126
```

In[∘]:=
```
ExploitationIndex[[1]]
```

Out[∘]=
```
{1.13126, 1.12932, 1.1277, 1.12792, 1.12706, 1.12472, 1.12291, 1.1259, 1.12363, 1.1163, 1.12266, 1.11992,
 1.12519, 1.11923, 1.11707, 1.11867, 1.11779, 1.11957, 1.11434, 1.1129, 1.11222, 1.11775, 1.11268, 1.11412,
 1.10174, 1.10931, 1.10606, 1.11387, 1.10673, 1.10096, 1.10092, 1.10745, 1.11029, 1.10942, 1.10541,
 1.10804, 1.09705, 1.11452, 1.11126, 1.10499, 1.09655, 1.11041, 1.09492, 1.10047, 1.10784, 1.07544,
 1.10148, 1.09044, 1.07418, 1.09419, 1.07248, 1.0884, 1.06894, 1.09979, 1.08886, 1.07368, 1.09169, 1.06904,
 1.08098, 1.08191, 1.08228, 1.04255, 1.08847, 1.06379, 1.03192, 1.06866, 1.0722, 1.07078, 1.00508, 1.0645,
 1.04588, 1.02569, 1.04224, 1.05334, 1.05761, 1.04877, 1.05666, 1.04736, 1.05306, 1.01342, 1.01564,
 1.00579, 0.998474, 1.00473, 0.995464, 0.97687, 1.02834, 0.958089, 1.00498, 1.0184, 1.01159, 0.987507,
 1.01139, 0.97179, 1.00546, 1.01519, 0.991293, 0.978243, 0.924488, 0.985054, 0.989182, 0.972381, 0.987957,
 0.980593, 0.946587, 0.894277, 0.942596, 0.940505, 0.86837, 0.932441, 0.870309, 0.890949, 0.828687,
 0.873182, 0.90564, 0.839551, 0.872043, 0.787669, 0.845302, 0.827494, 0.765814, 0.860349, 0.840132,
 0.865837, 0.848578, 0.789502, 0.796624, 0.814419, 0.848025, 0.821252, 0.848961, 0.804806, 0.76914,
 0.79925, 0.810348, 0.755043, 0.814088, 0.807332, 0.781905, 0.725534, 0.765803, 0.6471, 0.648065, 0.446642}
```

In[∘]:=
```
CountryList
```

Out[∘]=
```
{Myanmar, Rwanda, Mozambique, Yemen, Ethiopia, Mali, Burundi, Uganda, Malawi, Burkina Faso, Liberia,
 Central African Republic, Syria, Madagascar, Cambodia, Togo, Bangladesh, El Salvador, Ivory Coast,
 Sudan, Sierra Leone, Vietnam, Pakistan, Laos, Niger, Tanzania, Nepal, Kenya, Congo - Kinshasa,
 Gambia, Benin, India, Lesotho, Congo - Brazzaville, Cameroon, Egypt, Nigeria, Sri Lanka, Bolivia,
 Zimbabwe, Iraq, Belize, Honduras, Indonesia, Kyrgyzstan, Senegal, Fiji, Zambia, Haiti, China, Maldives,
 Ghana, Guatemala, Tajikistan, Philippines, Namibia, Bulgaria, Nicaragua, Jordan, Jamaica, Mongolia,
 Mauritania, Armenia, Dominican Republic, Morocco, Peru, Albania, Moldova, Angola, Panama, Paraguay,
 Eswatini, South Africa, Costa Rica, Barbados, Ecuador, Trinidad and Tobago, Botswana, Romania, Brazil,
 Colombia, Gabon, Turkey, Mauritius, Venezuela, Tunisia, Poland, Iran, Uruguay, Lithuania, Chile, Mexico,
 Kazakhstan, Thailand, Hungary, Estonia, Latvia, Argentina, Algeria, Serbia, Croatia, Malaysia, Ukraine,
 Slovakia, Malta, Saudi Arabia, Russia, South Korea, Macao, New Zealand, Bahrain, Taiwan, Qatar, Ireland,
 United Kingdom, Spain, Slovenia, Portugal, France, Singapore, Kuwait, Israel, Netherlands, Czech Republic,
 Denmark, Cyprus, Greece, Belgium, Canada, Finland, Germany, Austria, Italy, Sweden, Norway, Hong Kong,
 United States, Australia, Japan, Iceland, Switzerland, Brunei, Luxembourg, United Arab Emirates}
```

In[∘]:=
```
(ExploitTable = Table[{CountryList[[i]], ExploitationIndex[[1, i]], DataNoHeader[[i, 5]]}, {i, 1, N}]) // TableForm
Export["./ExploitTable2000.csv", ExploitTable, "CSV"]
```

Out[∘]//TableForm=

Myanmar	1.13126	845.849
Rwanda	1.12932	1046.39
Mozambique	1.1277	1072.96
Yemen	1.12792	1087.11
Ethiopia	1.12706	1214.7
Mali	1.12472	1465.29
Burundi	1.12291	1788.
Uganda	1.1259	1929.56
Malawi	1.12363	2210.34
Burkina Faso	1.1163	2378.16
Liberia	1.12266	2401.07
Central African Republic	1.11992	2538.35
Syria	1.12519	2658.67
Madagascar	1.11923	2914.76
Cambodia	1.11707	3291.12

Togo	1.11867	3334.02
Bangladesh	1.11779	3370.29
El Salvador	1.11957	3488.12
Ivory Coast	1.11434	3489.36
Sudan	1.1129	3615.44
Sierra Leone	1.11222	3652.33
Vietnam	1.11775	4061.21
Pakistan	1.11268	4081.11
Laos	1.11412	4167.76
Niger	1.10174	4346.08
Tanzania	1.10931	4520.93
Nepal	1.10606	4656.9
Kenya	1.11387	4835.72
Congo – Kinshasa	1.10673	5145.1
Gambia	1.10096	5190.17
Benin	1.10092	5617.26
India	1.10745	5755.65
Lesotho	1.11029	5878.64
Congo – Brazzaville	1.10942	5938.55
Cameroon	1.10541	6022.08
Egypt	1.10804	6212.78
Nigeria	1.09705	6470.12
Sri Lanka	1.11452	6944.46
Bolivia	1.11126	6958.61
Zimbabwe	1.10499	7265.56
Iraq	1.09655	8528.93
Belize	1.11041	8539.87
Honduras	1.09492	8813.38
Indonesia	1.10047	8817.81
Kyrgyzstan	1.10784	9170.95
Senegal	1.07544	9196.74
Fiji	1.10148	9490.13
Zambia	1.09044	10 765.5
Haiti	1.07418	10 790.3
China	1.09419	11 031.3
Maldives	1.07248	11 087.4

Ghana	1.0884	11 548.4
Guatemala	1.06894	12 883.9
Tajikistan	1.09979	13 014.9
Philippines	1.08886	13 194.6
Namibia	1.07368	14 451.1
Bulgaria	1.09169	14 688.
Nicaragua	1.06904	14 847.5
Jordan	1.08098	15 774.5
Jamaica	1.08191	15 996.7
Mongolia	1.08228	16 034.6
Mauritania	1.04255	16 563.5
Armenia	1.08847	16 863.7
Dominican Republic	1.06379	18 219.5
Morocco	1.03192	19 379.5
Peru	1.06866	20 437.1
Albania	1.0722	20 735.8
Moldova	1.07078	20 919.2
Angola	1.00508	20 954.4
Panama	1.0645	21 627.7
Paraguay	1.04588	22 090.2
Eswatini	1.02569	22 373.7
South Africa	1.04224	23 304.
Costa Rica	1.05334	23 635.7
Barbados	1.05761	24 699.4
Ecuador	1.04877	25 349.8
Trinidad and Tobago	1.05666	25 857.3
Botswana	1.04736	25 869.3
Romania	1.05306	28 743.7
Brazil	1.01342	30 691.3
Colombia	1.01564	31 804.
Gabon	1.00579	34 089.6
Turkey	0.998474	34 106.1
Mauritius	1.00473	37 010.1
Venezuela	0.995464	37 666.6
Tunisia	0.97687	37 809.9
Poland	1.02834	39 225.6

Iran	0.958089	39 246.8
Uruguay	1.00498	41 157.1
Lithuania	1.0184	41 622.2
Chile	1.01159	42 519.9
Mexico	0.987507	45 081.8
Kazakhstan	1.01139	46 317.5
Thailand	0.97179	46 956.6
Hungary	1.00546	47 703.4
Estonia	1.01519	48 759.3
Latvia	0.991293	51 229.3
Argentina	0.978243	53 129.4
Algeria	0.924488	53 442.2
Serbia	0.985054	53 459.3
Croatia	0.989182	53 561.7
Malaysia	0.972381	53 973.2
Ukraine	0.987957	57 068.
Slovakia	0.980593	63 912.3
Malta	0.946587	69 263.8
Saudi Arabia	0.894277	74 708.2
Russia	0.942596	80 560.9
South Korea	0.940505	82 232.3
Macao	0.86837	84 155.
New Zealand	0.932441	88 474.
Bahrain	0.870309	88 559.7
Taiwan	0.890949	93 078.8
Qatar	0.828687	100 278.
Ireland	0.873182	106 494.
United Kingdom	0.90564	110 602.
Spain	0.839551	115 253.
Slovenia	0.872043	122 388.
Portugal	0.787669	122 906.
France	0.845302	123 168.
Singapore	0.827494	125 657.
Kuwait	0.765814	126 076.
Israel	0.860349	131 540.
Netherlands	0.840132	137 070.

```
Czech Republic            0.865837   138 122.
Denmark                   0.848578   138 490.
Cyprus                    0.789502   138 600.
Greece                    0.796624   143 979.
Belgium                   0.814419   145 407.
Canada                    0.848025   147 381.
Finland                   0.821252   148 641.
Germany                   0.848961   149 345.
Austria                   0.804806   158 845.
Italy                     0.76914    163 692.
Sweden                    0.79925    166 011.
Norway                    0.810348   166 299.
Hong Kong                 0.755043   168 844.
United States             0.814088   175 243.
Australia                 0.807332   178 396.
Japan                     0.781905   187 632.
Iceland                   0.725534   197 848.
Switzerland               0.765803   211 198.
Brunei                    0.6471     244 230.
Luxembourg                0.648065   267 497.
United Arab Emirates      0.446642   498 721.
```

Out[•]= ./ExploitTable2000.csv

In[•]:=
```
(* ExploitationArray=ArrayPlot[Transpose[ExploitationIndex],
  FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./ExploitationArray.eps",ExploitationArray,"EPS"] *)
```

In[•]:=
```
ExploitationIndexDistrib = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndexDistrib[[t, i]] = {t, ExploitationIndex[[t, i]]}
 ]
]
```

Below the Gini coefficient for the exploitation intensity index is plotted over T. The pattern of the Gini coefficient is consistent with the previous charts depicting the exploitation intensity index.

In[∘]:=
```
(* Gini=Table[0.0,{T}];
For[t=1,t≤T,t++,
 Gini[[t]]=N/(N-1) ((Sum[Sum[Abs[ExploitationIndex[[t,i]]-ExploitationIndex[[t,j]]],{j,1,N}],{i,1,N}])/(2 N^2 Mean[ExploitationIndex[[t]]]))
] *)
```

In[∘]:=
```
Clear[SortExploitationIndex];
SortExploitationIndex = Table[Table[0.0, {N}], {t, 1, T}];
GiniTest = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortExploitationIndex[[t]] = Sort[ExploitationIndex[[t]]];
 GiniTest[[t]] = N/(N - 1) Sum[((2 i - N - 1) SortExploitationIndex[[t, i]]) / (N^2 Mean[SortExploitationIndex[[t]]]), {i, 1, N}]
]
```

In[∘]:=
```
Chop[GiniTest]
```

Out[∘]=
```
{0.0688573}
```

In[∘]:=
```
(* ExploitationGini=ListLinePlot[GiniTest,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: e_t^y"}] *)
```

In[∘]:=
```
(* Export["./ExploitationGini.eps",ExploitationGini,"EPS"] *)
```

Plotting Gini coefficient for wealth $W_{t-1} = p_{t-1}\,\omega_{t-1}$.

```
In[•]:= Clear[SortWealth, WealthGini];
SortWealth = Table[Table[0.0, {N}], {t, 1, T}];
WealthGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortWealth[[t]] = If[t == 1, Sort[AgentSet[[All, 1]]], Sort[Activities[[t - 1, All, 5]]]];
 WealthGini[[t]] = N/(N - 1) Sum[(2 i - N - 1) SortWealth[[t, i]]/(N^2 Mean[SortWealth[[t]]]), {i, 1, N}]
]
(* WealthGiniPlot=
 ListLinePlot[WealthGini,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: W^γ_{t-1}"},LabelStyle→12,PlotRange→{0,1}] *)

In[•]:= (* Export["./WealthGini.eps",WealthGiniPlot,"EPS"] *)

In[•]:= WealthGini

Out[•]= {0.83072}
```

The figure below plots the distribution of wealth for select *t*.

```
In[•]:= (* WealthDistribRow=GraphicsRow[{
    Histogram[AgentSet[[All,1]],10,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 1",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[24,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 25",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[49,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 50",18],AxesLabel→{"ω^γ_{t-1}"}]
   },Spacings→{15,-20}
  ]
  Export["./WealthDistribRow.eps",WealthDistribRow,"EPS"]
*)
```

Income Figures

Figures on net and gross income.

In[•]:=
```
Clear[Income];
Income = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  Income⟦t, i⟧ = Profit⟦t⟧ × Activities⟦t, i, 5⟧ + Wage⟦t⟧ × Activities⟦t, i, 6⟧
 ]
]

IncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
```

$$\text{IncomeGini}[[t]] = \frac{N}{N-1}\left(\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\text{Abs}[\text{Income}[[t, i]] - \text{Income}[[t, j]]]\right) \Big/ \left(2\,N^2\,\text{Mean}[\text{Income}[[t]]]\right)\right)$$

```
]
(* IncomeGiniPlot=ListLinePlot[IncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[IncomeGini]+0.1},AxesLabel→{"t","Gini: Net Income"},LabelStyle→14] *)
```

In[•]:=
```
(* Export["./NetIncomeGini.eps",IncomeGiniPlot,"EPS"] *)
```

```
In[•]:= Clear[IncomeShares];
IncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  IncomeShares[[t, i]] = Income[[t, i]] / Total[Income[[t]]]
 ]
]
(* IncomeArray=ArrayPlot[Transpose[IncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},
  PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",ColorFunctionScaling→True,
  DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./NetIncomeArray.eps",IncomeArray,"EPS"] *)

In[•]:= Clear[GrossIncome];
GrossIncome = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncome[[t, i]] = (1 + Profit[[t]]) Activities[[t, i, 5]] + Wage[[t]] × Activities[[t, i, 6]]
 ]
]

GrossIncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 GrossIncomeGini[[t]] = N/(N - 1) ((Sum[Sum[Abs[GrossIncome[[t, i]] - GrossIncome[[t, j]]], {j, 1, N}], {i, 1, N}]) / (2 N^2 Mean[GrossIncome[[t]]]))
]
(* GrossIncomeGiniPlot=ListLinePlot[GrossIncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[GrossIncomeGini]+0.1},AxesLabel→{"t","Gini: Income"},LabelStyle→14] *)
```

```
In[•]:= (* Export["./GrossIncomeGini.eps",GrossIncomeGiniPlot,"EPS"] *)

In[•]:= GrossIncomeGini

Out[•]= {0.809322}

In[•]:= Clear[GrossIncomeShares];
GrossIncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncomeShares[[t, i]] = GrossIncome[[t, i]] / Total[GrossIncome[[t]]]
 ]
]
(* GrossIncomeArray=
 ArrayPlot[Transpose[GrossIncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./GrossIncomeArray.eps",GrossIncomeArray,"EPS"] *)
```

Updated Simulation Reporting

```
In[•]:= (ExploitedTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploitedTable2000.csv", ExploitedTable, "CSV"]

Out[•]//TableForm=
Myanmar      &   1.13126   \\
Rwanda       &   1.12932   \\
Mozambique   &   1.1277    \\
Yemen        &   1.12792   \\
Ethiopia     &   1.12706   \\
Mali         &   1.12472   \\
Burundi      &   1.12291   \\
```

```
Uganda                    &   1.1259    \\
Malawi                    &   1.12363   \\
Burkina Faso              &   1.1163    \\
Liberia                   &   1.12266   \\
Central African Republic  &   1.11992   \\
Syria                     &   1.12519   \\
Madagascar                &   1.11923   \\
Cambodia                  &   1.11707   \\
Togo                      &   1.11867   \\
Bangladesh                &   1.11779   \\
El Salvador               &   1.11957   \\
Ivory Coast               &   1.11434   \\
Sudan                     &   1.1129    \\
Sierra Leone              &   1.11222   \\
Vietnam                   &   1.11775   \\
Pakistan                  &   1.11268   \\
Laos                      &   1.11412   \\
Niger                     &   1.10174   \\
Tanzania                  &   1.10931   \\
Nepal                     &   1.10606   \\
Kenya                     &   1.11387   \\
Congo - Kinshasa          &   1.10673   \\
Gambia                    &   1.10096   \\
Benin                     &   1.10092   \\
India                     &   1.10745   \\
Lesotho                   &   1.11029   \\
Congo - Brazzaville       &   1.10942   \\
Cameroon                  &   1.10541   \\
Egypt                     &   1.10804   \\
Nigeria                   &   1.09705   \\
Sri Lanka                 &   1.11452   \\
Bolivia                   &   1.11126   \\
Zimbabwe                  &   1.10499   \\
Iraq                      &   1.09655   \\
Belize                    &   1.11041   \\
Honduras                  &   1.09492   \\
```

```
Indonesia                 &   1.10047   \\
Kyrgyzstan                &   1.10784   \\
Senegal                   &   1.07544   \\
Fiji                      &   1.10148   \\
Zambia                    &   1.09044   \\
Haiti                     &   1.07418   \\
China                     &   1.09419   \\
Maldives                  &   1.07248   \\
Ghana                     &   1.0884    \\
Guatemala                 &   1.06894   \\
Tajikistan                &   1.09979   \\
Philippines               &   1.08886   \\
Namibia                   &   1.07368   \\
Bulgaria                  &   1.09169   \\
Nicaragua                 &   1.06904   \\
Jordan                    &   1.08098   \\
Jamaica                   &   1.08191   \\
Mongolia                  &   1.08228   \\
Mauritania                &   1.04255   \\
Armenia                   &   1.08847   \\
Dominican Republic        &   1.06379   \\
Morocco                   &   1.03192   \\
Peru                      &   1.06866   \\
Albania                   &   1.0722    \\
Moldova                   &   1.07078   \\
Angola                    &   1.00508   \\
Panama                    &   1.0645    \\
Paraguay                  &   1.04588   \\
Eswatini                  &   1.02569   \\
South Africa              &   1.04224   \\
Costa Rica                &   1.05334   \\
Barbados                  &   1.05761   \\
Ecuador                   &   1.04877   \\
Trinidad and Tobago       &   1.05666   \\
Botswana                  &   1.04736   \\
Romania                   &   1.05306   \\
```

```
Brazil                    &    1.01342    \\
Colombia                  &    1.01564    \\
Gabon                     &    1.00579    \\
Mauritius                 &    1.00473    \\
Poland                    &    1.02834    \\
Uruguay                   &    1.00498    \\
Lithuania                 &    1.0184     \\
Chile                     &    1.01159    \\
Kazakhstan                &    1.01139    \\
Hungary                   &    1.00546    \\
Estonia                   &    1.01519    \\
```

Out[•]= ./ExploitedTable2000.csv

In[•]:=
```
(ExploiterTable = DeleteCases[Table[If[ExploitationIndex⟦1, i⟧ < 1.0,
      {CountryList⟦i⟧, "&", ExploitationIndex⟦1, i⟧, "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploiterTable2000.csv", ExploiterTable, "CSV"]
```

Out[•]//TableForm=
```
Turkey                    &    0.998474    \\
Venezuela                 &    0.995464    \\
Tunisia                   &    0.97687     \\
Iran                      &    0.958089    \\
Mexico                    &    0.987507    \\
Thailand                  &    0.97179     \\
Latvia                    &    0.991293    \\
Argentina                 &    0.978243    \\
Algeria                   &    0.924488    \\
Serbia                    &    0.985054    \\
Croatia                   &    0.989182    \\
Malaysia                  &    0.972381    \\
Ukraine                   &    0.987957    \\
Slovakia                  &    0.980593    \\
Malta                     &    0.946587    \\
Saudi Arabia              &    0.894277    \\
Russia                    &    0.942596    \\
South Korea               &    0.940505    \\
Macao                     &    0.86837     \\
```

```
New Zealand            &   0.932441   \\
Bahrain                &   0.870309   \\
Taiwan                 &   0.890949   \\
Qatar                  &   0.828687   \\
Ireland                &   0.873182   \\
United Kingdom         &   0.90564    \\
Spain                  &   0.839551   \\
Slovenia               &   0.872043   \\
Portugal               &   0.787669   \\
France                 &   0.845302   \\
Singapore              &   0.827494   \\
Kuwait                 &   0.765814   \\
Israel                 &   0.860349   \\
Netherlands            &   0.840132   \\
Czech Republic         &   0.865837   \\
Denmark                &   0.848578   \\
Cyprus                 &   0.789502   \\
Greece                 &   0.796624   \\
Belgium                &   0.814419   \\
Canada                 &   0.848025   \\
Finland                &   0.821252   \\
Germany                &   0.848961   \\
Austria                &   0.804806   \\
Italy                  &   0.76914    \\
Sweden                 &   0.79925    \\
Norway                 &   0.810348   \\
Hong Kong              &   0.755043   \\
United States          &   0.814088   \\
Australia              &   0.807332   \\
Japan                  &   0.781905   \\
Iceland                &   0.725534   \\
Switzerland            &   0.765803   \\
Brunei                 &   0.6471     \\
Luxembourg             &   0.648065   \\
United Arab Emirates   &   0.446642   \\
```

Out[•]= ./ExploiterTable2000.csv

In[•]:=
```
(ExploitIncomeCSV = Table[{CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"}, {i, 1, N}]) // TableForm
Export["./ExploitIncomeTable2000.csv", ExploitIncomeCSV, "CSV"]
```

Out[•]//TableForm=

Myanmar	&	1.13126	\\
Rwanda	&	1.12932	\\
Mozambique	&	1.1277	\\
Yemen	&	1.12792	\\
Ethiopia	&	1.12706	\\
Mali	&	1.12472	\\
Burundi	&	1.12291	\\
Uganda	&	1.1259	\\
Malawi	&	1.12363	\\
Burkina Faso	&	1.1163	\\
Liberia	&	1.12266	\\
Central African Republic	&	1.11992	\\
Syria	&	1.12519	\\
Madagascar	&	1.11923	\\
Cambodia	&	1.11707	\\
Togo	&	1.11867	\\
Bangladesh	&	1.11779	\\
El Salvador	&	1.11957	\\
Ivory Coast	&	1.11434	\\
Sudan	&	1.1129	\\
Sierra Leone	&	1.11222	\\
Vietnam	&	1.11775	\\
Pakistan	&	1.11268	\\
Laos	&	1.11412	\\
Niger	&	1.10174	\\
Tanzania	&	1.10931	\\
Nepal	&	1.10606	\\
Kenya	&	1.11387	\\
Congo - Kinshasa	&	1.10673	\\
Gambia	&	1.10096	\\
Benin	&	1.10092	\\

```
India                    &   1.10745    \\
Lesotho                  &   1.11029    \\
Congo – Brazzaville      &   1.10942    \\
Cameroon                 &   1.10541    \\
Egypt                    &   1.10804    \\
Nigeria                  &   1.09705    \\
Sri Lanka                &   1.11452    \\
Bolivia                  &   1.11126    \\
Zimbabwe                 &   1.10499    \\
Iraq                     &   1.09655    \\
Belize                   &   1.11041    \\
Honduras                 &   1.09492    \\
Indonesia                &   1.10047    \\
Kyrgyzstan               &   1.10784    \\
Senegal                  &   1.07544    \\
Fiji                     &   1.10148    \\
Zambia                   &   1.09044    \\
Haiti                    &   1.07418    \\
China                    &   1.09419    \\
Maldives                 &   1.07248    \\
Ghana                    &   1.0884     \\
Guatemala                &   1.06894    \\
Tajikistan               &   1.09979    \\
Philippines              &   1.08886    \\
Namibia                  &   1.07368    \\
Bulgaria                 &   1.09169    \\
Nicaragua                &   1.06904    \\
Jordan                   &   1.08098    \\
Jamaica                  &   1.08191    \\
Mongolia                 &   1.08228    \\
Mauritania               &   1.04255    \\
Armenia                  &   1.08847    \\
Dominican Republic       &   1.06379    \\
Morocco                  &   1.03192    \\
Peru                     &   1.06866    \\
Albania                  &   1.0722     \\
```

```
Moldova                    &   1.07078    \\
Angola                     &   1.00508    \\
Panama                     &   1.0645     \\
Paraguay                   &   1.04588    \\
Eswatini                   &   1.02569    \\
South Africa               &   1.04224    \\
Costa Rica                 &   1.05334    \\
Barbados                   &   1.05761    \\
Ecuador                    &   1.04877    \\
Trinidad and Tobago        &   1.05666    \\
Botswana                   &   1.04736    \\
Romania                    &   1.05306    \\
Brazil                     &   1.01342    \\
Colombia                   &   1.01564    \\
Gabon                      &   1.00579    \\
Turkey                     &   0.998474   \\
Mauritius                  &   1.00473    \\
Venezuela                  &   0.995464   \\
Tunisia                    &   0.97687    \\
Poland                     &   1.02834    \\
Iran                       &   0.958089   \\
Uruguay                    &   1.00498    \\
Lithuania                  &   1.0184     \\
Chile                      &   1.01159    \\
Mexico                     &   0.987507   \\
Kazakhstan                 &   1.01139    \\
Thailand                   &   0.97179    \\
Hungary                    &   1.00546    \\
Estonia                    &   1.01519    \\
Latvia                     &   0.991293   \\
Argentina                  &   0.978243   \\
Algeria                    &   0.924488   \\
Serbia                     &   0.985054   \\
Croatia                    &   0.989182   \\
Malaysia                   &   0.972381   \\
Ukraine                    &   0.987957   \\
```

```
Slovakia                  &   0.980593   \\
Malta                     &   0.946587   \\
Saudi Arabia              &   0.894277   \\
Russia                    &   0.942596   \\
South Korea               &   0.940505   \\
Macao                     &   0.86837    \\
New Zealand               &   0.932441   \\
Bahrain                   &   0.870309   \\
Taiwan                    &   0.890949   \\
Qatar                     &   0.828687   \\
Ireland                   &   0.873182   \\
United Kingdom            &   0.90564    \\
Spain                     &   0.839551   \\
Slovenia                  &   0.872043   \\
Portugal                  &   0.787669   \\
France                    &   0.845302   \\
Singapore                 &   0.827494   \\
Kuwait                    &   0.765814   \\
Israel                    &   0.860349   \\
Netherlands               &   0.840132   \\
Czech Republic            &   0.865837   \\
Denmark                   &   0.848578   \\
Cyprus                    &   0.789502   \\
Greece                    &   0.796624   \\
Belgium                   &   0.814419   \\
Canada                    &   0.848025   \\
Finland                   &   0.821252   \\
Germany                   &   0.848961   \\
Austria                   &   0.804806   \\
Italy                     &   0.76914    \\
Sweden                    &   0.79925    \\
Norway                    &   0.810348   \\
Hong Kong                 &   0.755043   \\
United States             &   0.814088   \\
Australia                 &   0.807332   \\
Japan                     &   0.781905   \\
```

```
Iceland                    &    0.725534    \\
Switzerland                &    0.765803    \\
Brunei                     &    0.6471      \\
Luxembourg                 &    0.648065    \\
United Arab Emirates       &    0.446642    \\
```

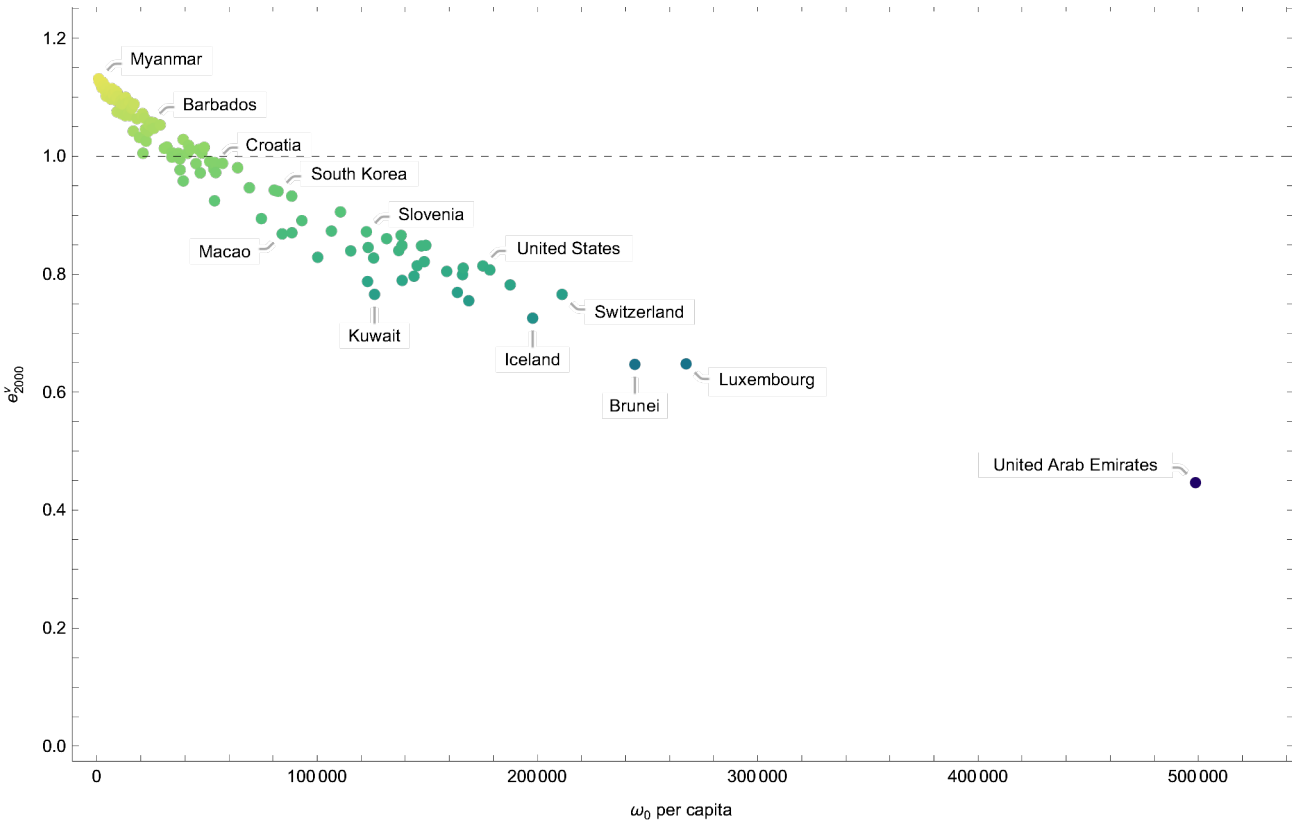
Out[•]= ./ExploitIncomeTable2000.csv

In[•]:= **DataNoHeader[[1]]**

Out[•]= {Myanmar, 39 517.8, 1.5157, 46.7197, 845.849}

In[∘]:=
```
ExploitWealthPlot = ListPlot[Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, 1, N}] → CountryList,
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_2000"}, ColorFunction → "BlueGreenYellow",
  Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitWealthPlot2000.eps", ExploitWealthPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploitWealthPlot2000.eps

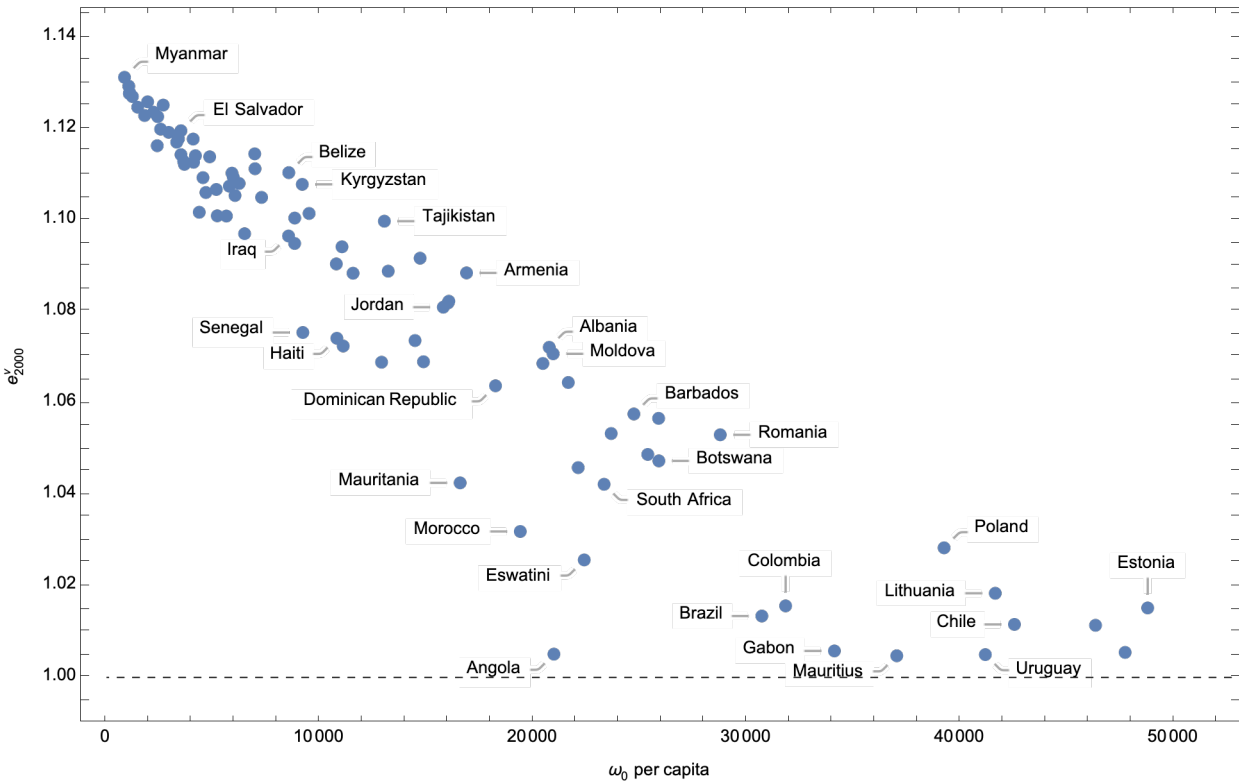
In[ ]:=
```
ExploitedCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1, i, 0.], {i, 1, N}], 0.]
ExploiterCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1, i, 0.], {i, 1, N}], 0.]
```

Out[ ]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32,
 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61,
 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 84, 87, 89, 90, 91, 93, 95, 96}
```

Out[ ]=
```
{83, 85, 86, 88, 92, 94, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108,
 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126,
 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144}
```

In[∘]:=
```
ExploitedCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploitedCountries}] → CountryList[[ExploitedCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_2000"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitedCountriesPlot2000.eps", ExploitedCountriesPlot, "EPS"]
```

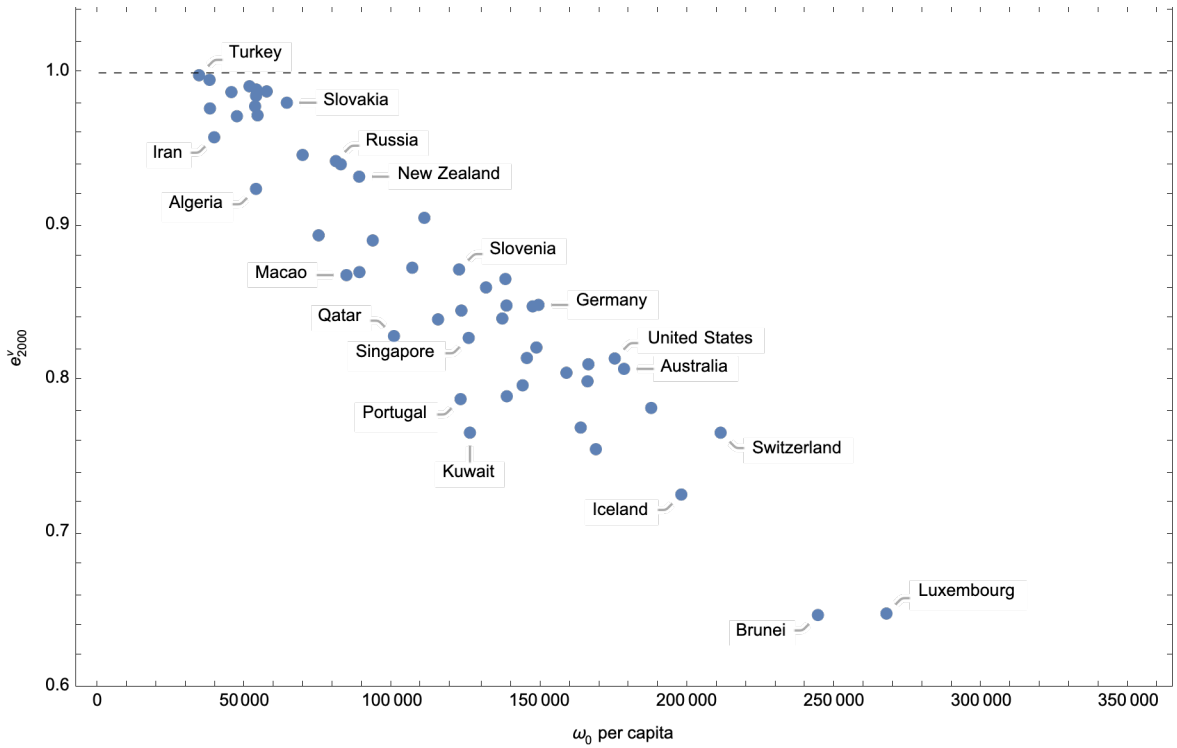
Out[∘]=



Out[∘]= ./ExploitedCountriesPlot2000.eps

In[∘]:=
```
ExploiterCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploiterCountries}] → CountryList[[ExploiterCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → {0.6, 1.0}, Frame → True,
  FrameLabel → {"ω0 per capita", "e^ν_2000"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploiterCountriesPlot2000.eps", ExploiterCountriesPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploiterCountriesPlot2000.eps

In[∘]:=
```
(outlierTesting = Table[{CountryList[[i]], AgentSet[[i, 1]], AgentSet[[i, 2]]}, {i, 1, Length[CountryList]}]) // TableForm
Export["./outlierTesting2000.csv", outlierTesting, "CSV"];
```

Out[∘]//TableForm=

Myanmar	39 517.8	7.0813×10^6
Rwanda	8301.77	1.07597×10^6

Mozambique	19 004.2	2.00038×10^6
Yemen	18 925.7	2.04348×10^6
Ethiopia	80 443.1	7.8781×10^6
Mali	16 039.6	1.25282×10^6
Burundi	11 405.5	769 550.
Uganda	45 634.5	3.96874×10^6
Malawi	24 642.7	1.75805×10^6
Burkina Faso	27 605.4	1.24146×10^6
Liberia	6839.45	452 883.
Central African Republic	9240.62	508 782.
Syria	43 631.	3.55031×10^6
Madagascar	45 956.4	2.42635×10^6
Cambodia	40 004.2	1.87251×10^6
Togo	16 418.	839 019.
Bangladesh	430 244.	2.09346×10^7
El Salvador	20 537.7	1.1071×10^6
Ivory Coast	57 416.3	2.34841×10^6
Sudan	98 611.1	3.78141×10^6
Sierra Leone	16 744.4	623 643.
Vietnam	324 533.	1.57583×10^7
Pakistan	580 920.	2.20647×10^7
Laos	22 187.9	898 215.
Niger	49 248.	1.26687×10^6
Tanzania	147 306.	4.88357×10^6
Nepal	111 491.	3.28951×10^6
Kenya	154 572.	6.18803×10^6
Congo – Kinshasa	242 364.	7.31647×10^6
Gambia	6839.09	171 947.
Benin	38 568.1	968 312.
India	6.08128×10^6	1.88292×10^8
Lesotho	11 950.1	411 439.
Congo – Brazzaville	18 572.2	618 318.
Cameroon	93 426.	2.6974×10^6

Egypt	427 636.	1.35261×10^{7}
Nigeria	791 192	1.78461×10^{7}
Sri Lanka	130 400.	5.37809×10^{6}
Bolivia	58 579.6	2.09666×10^{6}
Zimbabwe	86 325.7	2.45804×10^{6}
Iraq	200 409.	4.46196×10^{6}
Belize	2111.91	73 042.5
Honduras	57 943.6	1.23745×10^{6}
Indonesia	1 865 089	4.62306×10^{7}
Kyrgyzstan	45 127.5	1.41706×10^{6}
Senegal	90 106.9	1.28585×10^{6}
Fiji	7696.5	196 449.
Zambia	112 132.	2.15245×10^{6}
Haiti	91 326.6	1.27524×10^{6}
China	14 236 407	2.98569×10^{8}
Maldives	3097.83	42 044.1
Ghana	222 640.	4.08462×10^{6}
Guatemala	150 106.	1.92388×10^{6}
Tajikistan	80 904.2	1.96659×10^{6}
Philippines	1.02907×10^{6}	1.90667×10^{7}
Namibia	25 933.9	359 099.
Bulgaria	117 474.	2.32086×10^{6}
Nicaragua	75 266.4	966 209.
Jordan	80 804.9	1.27489×10^{6}
Jamaica	42 466.4	682 099.
Mongolia	38 441.3	621 838.
Mauritania	43 565.2	391 374.
Armenia	51 764.7	951 146.
Dominican Republic	154 343.	1.82878×10^{6}
Morocco	558 006.	4.45669×10^{6}
Peru	540 764.	6.90074×10^{6}
Albania	64 886.5	876 520.
Moldova	87 916.9	1.16045×10^{6}

Angola	343 558.	2.12633×10^6
Panama	65 538.5	784 817.
Paraguay	117 590.	1.09881×10^6
Eswatini	22 494.5	168 535.
South Africa	1.04793×10^6	9.38026×10^6
Costa Rica	93 654.1	960 565.
Barbados	6705.88	72 802.7
Ecuador	321 464.	3.11143×10^6
Trinidad and Tobago	32 766.4	351 166.
Botswana	42 511.	404 400.
Romania	636 311.	6.50286×10^6
Brazil	5 364 543	3.57481×10^7
Colombia	1 260 394	8.57197×10^6
Gabon	41 875.7	260 752.
Turkey	2 156 879	1.26259×10^7
Mauritius	43 860.6	270 629.
Venezuela	911 244.	5.20451×10^6
Tunisia	367 074.	1.81751×10^6
Poland	1 512 410	1.16391×10^7
Iran	2.57551×10^6	1.11894×10^7
Uruguay	136 629.	844 864.
Lithuania	145 752.	1.0172×10^6
Chile	652 358	4.27547×10^6
Mexico	4 458 584	2.3911×10^7
Kazakhstan	691 182.	4.52203×10^6
Thailand	2.95604×10^6	1.41108×10^7
Hungary	487 552.	3.02741×10^6
Estonia	68 219.2	462 039.
Latvia	122 141.	674 705.
Argentina	1.95892×10^6	9.79731×10^6
Algeria	1.65896×10^6	5.85177×10^6
Serbia	401 816.	2.11471×10^6
Croatia	237 176.	1.28862×10^6

Malaysia	1 251 871	6.00106×10^6
Ukraine	2.78709×10^6	1.49992×10^7
Slovakia	345 075.	1.75609×10^6
Malta	27 262.2	109 929.
Saudi Arabia	1.54376×10^6	4.61009×10^6
Russia	11 794 515	4.63825×10^7
South Korea	3.8961×10^6	1.51244×10^7
Macao	36 001.5	94 304.2
New Zealand	341 421	1.26205×10^6
Bahrain	58 856.8	155 643.
Taiwan	2.06491×10^6	6.06014×10^6
Qatar	59 414.9	129 366.
Ireland	402 878.	1.08057×10^6
United Kingdom	6 517 053	2.06797×10^7
Spain	4.70519×10^6	1.0759×10^7
Slovenia	243 271.	648 825.
Portugal	1.26557×10^6	2.31108×10^6
France	7 497 759	1.76031×10^7
Singapore	506 261.	1.09647×10^6
Kuwait	257 839.	430 780.
Israel	782 138.	1.97086×10^6
Netherlands	2 183 005	5.00497×10^6
Czech Republic	1.42119×10^6	3.67712×10^6
Denmark	739 703.	1.76324×10^6
Cyprus	96 133.3	176 889.
Greece	1.59559×10^6	3.02453×10^6
Belgium	1.49508×10^6	3.05746×10^6
Canada	4 508 144	1.07185×10^7
Finland	771 148.	1.62473×10^6
Germany	12 156 833	2.903×10^7
Austria	1.28177×10^6	2.5152×10^6
Italy	9 280 035	1.57128×10^7

Sweden	1.47444×10^6	2.82595×10^6
Norway	748 248.	1.50352×10^6
Hong Kong	1.11544×10^6	1.78548×10^6
United States	49 367 780	1.00813×10^8
Australia	3.388×10^6	6.7203×10^6
Japan	23 927 642	4.26696×10^7
Iceland	55 476.5	79 201.8
Switzerland	1.50875×10^6	2.52062×10^6
Brunei	81 377.3	87 008.5
Luxembourg	116 655.	125 161.
United Arab Emirates	1.56304×10^6	818 251.

2010

2010 - Model Setup

2010 - Basic Model

In[ ]:=
```
Clear[Activities, i, Wage, Profit, u];
(Activities = Table[Table[{0.0, 0.0, 0.0, 0.0, 0.0, 0.0}, {N}], {T}]);
Wage = Table[0.0, {T}];
Profit = Table[0.0, {T}];

(* First Stage t=1 simulation *)
(* Setting w1 and π1 *)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]], Wage[[1]] = b, Wage[[1]] = RandomReal[{b, (1 - A (1 + 0))/L}]],
```

```
  Wage[[1]] = (1 - A (1 + 0))/L];
Profit[[1]] = ((p - p0 A - Wage[[1]] L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^ν,y_t^ν,z_t^ν,δ_t^ν) for all ν and t=1*)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]],
   CapitalConstrained[1, A, L, l, 1, Profit[[1]], b], KnifeEdge[1, A, L, l, 1, Profit[[1]], b]],
  LabourConstrained[1, A, L, l, 1, Profit[[1]], b]
 ];

(* Simulation for remaining T-1 time steps *)
For[t = 2, t ≤ T, t++,

 (* Updating w_t *)
 If[Total[l] ≥ L A^-1 Total[Activities[[t - 1, All, 5]]], If[Total[l] > L A^-1 Total[Activities[[t - 1, All, 5]]],
   Wage[[t]] = b, Wage[[t]] = RandomReal[{b, ((1 - A (1 + r))/L /. r → 0)}]], Wage[[t]] = ((1 - A (1 + r))/L /. r → 0)];

 (* Updating r_t *)
 Profit[[t]] = ((p - p0 A - Wage[[t]] L)/(p0 A)) /. {p → 1, p0 → 1};

 (* Checking whether the simulation is capital constrained, labour constrained,
 or on the knife-edge and calling the corresponding function to find optimal (x_t^ν,y_t^ν,z_t^ν,δ_t^ν) for all ν *)
 If[Total[l] ≥ L A^-1 Total[Activities[[t - 1, All, 5]]],
  If[Total[l] > L A^-1 Total[Activities[[t - 1, All, 5]]],
   CapitalConstrained[1, A, L, l, t, Profit[[t]], b], KnifeEdge[1, A, L, l, t, Profit[[t]], b]],
```

```
  LabourConstrained[1, A, L, l, t, Profit⟦t⟧, b]
 ];

]
```

Exploitation and Class Over Time

The chart below captures the number of agents who are exploiters, exploited, or neither over the course of the simulation according to Definition 2, which defines exploitation in relation to Λ_t^v and $v_t\, c_t^v$, where v_t is just the embodied labour value and $c_t^v = \pi_t\, W_{t-1}^v + (w_t - b_t)\; l^v + b_t\, \Lambda_t^v$. An agent v is exploited during t if and only if $\Lambda_t^v > v_t\, c_t^v$, an agent is an exploiter if and only if $\Lambda_t^v < v_t\, c_t^v$, and an agent is neither exploited nor an exploiter if and only if $\Lambda_t^v = v_t\, c_t^v$.

In[∘]:=
```
Clear[IndivExploitation, ConsumptionBundle];
ConsumptionBundle = Table[0.0, {i, T}, {j, N}];
IndivExploitation = Table[0.0, {i, T}, {j, N}];

For[i = 1, i ≤ N, i++,
 ConsumptionBundle⟦1, i⟧ = EmbodiedValues⟦1, 2⟧ (Profit⟦1⟧ × AgentSet⟦i, 1⟧ + (Wage⟦1⟧ - b) l⟦i⟧ + b Activities⟦1, i, 6⟧);
 IndivExploitation⟦1, i⟧ = Activities⟦1, i, 6⟧ - ConsumptionBundle⟦1, i⟧;

]

For[t = 2, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ConsumptionBundle⟦t, i⟧ =
   EmbodiedValues⟦1, 2⟧ (Profit⟦t⟧ × Activities⟦t - 1, i, 5⟧ + (Wage⟦t⟧ - b) l⟦i⟧ + b Activities⟦t, i, 6⟧);
  IndivExploitation⟦t, i⟧ = Activities⟦t, i, 6⟧ - ConsumptionBundle⟦t, i⟧;
 ]
]

Clear[Exploiters, Exploited, NonExploit];
Exploiters = Table[0.0, {T}];
```

```
Exploited = Table[0.0, {T}];
NonExploit = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  If[IndivExploitation⟦t, i⟧ < 0, Exploiters⟦t⟧ ++, If[IndivExploitation⟦t, i⟧ == 0, NonExploit⟦t⟧ ++, Exploited⟦t⟧ ++]];
 ]
]

ExploitationLegend =
  Grid[{{Row[{Graphics[{Thick, Blue, Line[{{0, 0}, {1, 0}}]}], AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploiters", 20], FontFamily → "Times"]}], Spacer[10],
      Row[{Graphics[{Thick, Red, Dotted, Line[{{0, 0}, {1, 0}}]}], AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Neither Exploiter or Exploited", 20], FontFamily → "Times"]}]},
    {Row[{Graphics[{Thick, Black, Dashed, Line[{{0, 0}, {1, 0}}]}], AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploited", 20], FontFamily → "Times"]}], Spacer[10],
    }}, Frame → True, Alignment → Left];
(* ExploitationPlot=Labeled[
  ListLinePlot[{Exploiters,Exploited,NonExploit},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black},{Thick,Dotted,Red}},
   Frame→True,FrameLabel→{"t","Total ν in Group"},LabelStyle→20,
   PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],ExploitationLegend] *)
```

In[•]:=
```
Export["./ExploitationPlot.eps", ExploitationPlot, "EPS"]
```

Out[•]=
```
./ExploitationPlot.eps
```

The chart below captures the class composition of the simulation over time according to Corollary 1 of Theorem 3 (class is defined using labour endowments l^{ν} in relation to means of production).

In[•]:=
```
Clear[Class1, Class2, Class3, Class4, j, k];
Class1 = Table[0.0, {T}];
Class2 = Table[0.0, {T}];
Class3 = Table[0.0, {T}];
Class4 = Table[0.0, {T}];
```

```
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ == Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class2⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class3⟦j⟧++, 0];
 ]
]

Clear[k];
For[k = 1, k ≤ N, k++,
 If[AgentSet⟦k, 1⟧ == 0 && Profit⟦1⟧ > 0, Class4⟦1⟧++, 0];
]

Clear[j, k];
For[j = 2, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[Activities⟦j - 1, k, 5⟧ == 0 && Profit⟦j⟧ > 0, Class4⟦j⟧++, 0];
 ]
]
```

```
(* ClassLegend=
 Column[{
   Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^1 = {∑ν ∈ (+,0,+)\(+,0,0); A_t y_t^ν < z_t^ν}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,DotDashed,Green,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^2 = {∑ν ∈ (+,0,0); A_t y_t^ν = z_t^ν}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Purple,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^3 = {∑ν ∈ (+,+,0)\(+,0,0); A_t y_t^ν > z_t^ν}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Black,Dotted,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^4 = {∑ν ∈ (0,+,0); W_{t-1}^ν = 0}",20],FontFamily→"Times"]}]
  },Frame→True,Alignment→Left];
ClassPlotCorollary1=Labeled[
  ListLinePlot[{Class1,Class2,Class3,Class4},PlotStyle→{{Thick,Blue},{Thick,DotDashed,Green},
     {Thick,Dashed,Purple},{Thick, Dotted,Black},{Thick,DotDashed,Orange},{Thick,Dashed,Green}},
   PlotRange→{{1,T},{-10,N+10}},Frame→True,FrameLabel→{"t","Total ν in Classes"},
   LabelStyle→20,ImageSize→{500,350},AxesOrigin→{1,-10}],ClassLegend] *)
```

In[∘]:=
```
(* Export["./ClassPlotCorollary1.eps",ClassPlotCorollary1,"EPS"] *)
```

The chart below captures the intersection of class and exploitation over the simulation. If an agent is in C^1 according to Corollary 1 of Theorem 3 and is also an exploiter according to Definition 2, then the agent is in the group "$C^1 \wedge$ Exploiter". If an agent is in $C^3 \cup C^4$ and is exploited they are in group "$(C^3 \cup C^4) \wedge$ Exploited".

In[∘]:=
```
Clear[CECP1, CECP2, CECP3, j, k];
CECP1 = Table[0.0, {T}];
CECP2 = Table[0.0, {T}];
CECP3a = Table[0.0, {T}];
CECP3b = Table[0.0, {T}];
CECP3 = Table[0.0, {T}];
```

```
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ < 0, CECP1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
  If[If[j == 1, AgentSet⟦k, 1⟧, Activities⟦j - 1, k, 5⟧] == 0 && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
 ]
]


(* CECPLegend=
 Grid[{{Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],Spacer[5],
      Style[Style["∑v ∈ C_t^1 ∧ Exploiter",20],FontFamily→"Times"]}],Spacer[10],
    Row[{Graphics[{Thick,Black,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
      Spacer[5],Style[Style["∑v ∈ (C_t^3 ⋃ C_t^4) ∧ Exploited",20],FontFamily→"Times"]}],
   }},Frame→True,Alignment→Left];
CECPPlotCorollary1=
 Labeled[ListLinePlot[{CECP1,CECP3},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black}},Frame→True,FrameLabel→
    {"t","Total v in Group"},LabelStyle→20,PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],CECPLegend] *)
```

In[•]:=
```
(* Export["./CECPPlotCorollary1.eps",CECPPlotCorollary1,"EPS"] *)
```

The charts below display the distribution of an index of the intensity of exploitation across the agents over the simulation. The index itself is calculated as $e_t^v = \Lambda_t^v / v_t\, c_t^v$ and the charts that follow explore some possibilities for visualizing the intensity of exploitation over time.

In[∘]:=
```
Clear[ExploitationIndex];
ExploitationIndex = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndex[[t, i]] = Activities[[t, i, 6]] / (EmbodiedValues[[1, 2]] ConsumptionBundle[[t, i]] / EmbodiedValues[[1, 2]]) /. p → 1
 ]
]
```

The chart below uses *Mathematica's* `ArrayPlot` function to display the exploitation index of the N agents over *T*. There is a fixed distribution of uneven exploitation intensity while the simulation is capital constrained, however, this pattern disappears once the simulation is labour constrained, at which point exploitation intensity is the same for all $v \in \mathcal{N}$.

In[∘]:=
```
ExploitationIndex[[1, 1]]
```

Out[∘]=
```
1.12857
```

In[∘]:=
```
ExploitationIndex[[1]]
```

Out[∘]=
```
{1.12857, 1.1257, 1.12456, 1.12625, 1.12622, 1.12594, 1.12511, 1.12397, 1.12147, 1.11742, 1.11753, 1.1097,
 1.11719, 1.11779, 1.1219, 1.11533, 1.11416, 1.11083, 1.11638, 1.12347, 1.11048, 1.1167, 1.10851, 1.11151,
 1.10225, 1.10922, 1.1087, 1.09325, 1.09116, 1.09345, 1.10967, 1.10054, 1.09406, 1.10338, 1.08376, 1.09216,
 1.09138, 1.09644, 1.10607, 1.08419, 1.08695, 1.10213, 1.06761, 1.08978, 1.09315, 1.09964, 1.08543, 1.08069,
 1.08831, 1.09217, 1.05986, 1.05738, 1.06629, 1.07235, 1.07503, 1.04228, 1.07556, 1.04037, 1.02061, 1.04718,
 1.0154, 0.98797, 1.03835, 1.05167, 1.04885, 1.03659, 1.04088, 1.03636, 1.03462, 0.991675, 1.03194,
 1.03659, 1.04661, 1.03427, 1.04071, 1.03927, 1.02889, 1.0333, 1.00868, 0.98585, 0.991098, 0.964614,
 0.996299, 1.01403, 1.00795, 0.969502, 0.983665, 1.00011, 0.993939, 0.984499, 0.985189, 0.974677, 0.952756,
 0.990722, 0.985732, 0.989064, 0.957608, 0.97558, 0.971466, 0.982471, 0.963823, 0.94932, 0.953964, 0.95251,
 0.960433, 0.929153, 0.946738, 0.853304, 0.907326, 0.868684, 0.914693, 0.88944, 0.864406, 0.894342,
 0.88851, 0.889037, 0.780323, 0.840642, 0.870029, 0.879124, 0.869067, 0.829663, 0.794408, 0.817322,
 0.792838, 0.832247, 0.74826, 0.831309, 0.840325, 0.799052, 0.830046, 0.799664, 0.816258, 0.753365,
 0.807095, 0.778443, 0.7993, 0.767734, 0.784399, 0.704891, 0.702262, 0.725094, 0.650793, 0.682636}
```

In[∘]:=
```
CountryList
```

Out[∘]=
```
{Burundi, Mozambique, Mali, Rwanda, Malawi, Liberia, Congo - Kinshasa, Sierra Leone, Ethiopia, Burkina Faso,
 Central African Republic, Niger, Madagascar, Myanmar, Zimbabwe, Togo, Cambodia, Ivory Coast, Uganda,
 Kyrgyzstan, Benin, Kenya, Nepal, Pakistan, Gambia, Bangladesh, Cameroon, Yemen, Senegal, Tanzania, Bolivia,
 Lesotho, Laos, Vietnam, Haiti, El Salvador, Honduras, Egypt, Belize, India, Nicaragua, Moldova, Sudan, Syria,
 Fiji, Armenia, Zambia, Iraq, Philippines, Sri Lanka, Mauritania, Guatemala, Congo - Brazzaville, Ghana, Peru,
 Nigeria, Tajikistan, Namibia, Morocco, China, Maldives, Angola, Paraguay, Jordan, Panama, Colombia, Costa Rica,
 Dominican Republic, Indonesia, Eswatini, South Africa, Ecuador, Bulgaria, Mongolia, Kazakhstan, Ukraine, Albania,
 Poland, Jamaica, Tunisia, Brazil, Algeria, Mexico, Romania, Chile, Turkey, Mauritius, Argentina, Botswana,
 Thailand, Uruguay, Gabon, Iran, Malaysia, Barbados, Serbia, Venezuela, Lithuania, Russia, Slovakia, Hungary,
 Malta, Croatia, New Zealand, Israel, Trinidad and Tobago, Estonia, Kuwait, Latvia, Saudi Arabia, South Korea,
 Slovenia, Taiwan, Canada, Czech Republic, United Kingdom, Bahrain, Greece, Australia, United States, Germany,
 Iceland, Cyprus, France, Spain, Finland, Portugal, Sweden, Japan, Italy, Denmark, Ireland, Austria, Macao,
 Netherlands, Belgium, Norway, Hong Kong, Switzerland, Brunei, United Arab Emirates, Singapore, Qatar, Luxembourg}
```

In[∘]:=
```
(ExploitTable = Table[{CountryList[[i]], ExploitationIndex[[1, i]], DataNoHeader[[i, 5]]}, {i, 1, N}]) // TableForm
Export["./ExploitTable2010.csv", ExploitTable, "CSV"]
```

Out[∘]//TableForm=

Burundi	1.12857	1635.45
Mozambique	1.1257	2040.85
Mali	1.12456	2340.31
Rwanda	1.12625	2613.62
Malawi	1.12622	2780.84
Liberia	1.12594	2911.05
Congo - Kinshasa	1.12511	2960.46
Sierra Leone	1.12397	3029.84
Ethiopia	1.12147	3104.46
Burkina Faso	1.11742	3606.29
Central African Republic	1.11753	4512.85
Niger	1.1097	5116.06
Madagascar	1.11719	5138.93
Myanmar	1.11779	5228.28
Zimbabwe	1.1219	5580.72
Togo	1.11533	6075.33

Cambodia	1.11416	6272.68
Ivory Coast	1.11083	6415.49
Uganda	1.11638	6438.71
Kyrgyzstan	1.12347	6610.42
Benin	1.11048	6617.92
Kenya	1.1167	6957.15
Nepal	1.10851	7146.08
Pakistan	1.11151	7297.08
Gambia	1.10225	8233.84
Bangladesh	1.10922	8363.49
Cameroon	1.1087	8509.97
Yemen	1.09325	10 187.9
Senegal	1.09116	10 984.6
Tanzania	1.09345	11 426.5
Bolivia	1.10967	11 927.7
Lesotho	1.10054	11 961.5
Laos	1.09406	12 771.8
Vietnam	1.10338	13 201.9
Haiti	1.08376	14 282.6
El Salvador	1.09216	14 901.9
Honduras	1.09138	15 533.9
Egypt	1.09644	15 649.3
Belize	1.10607	16 323.6
India	1.08419	17 258.4
Nicaragua	1.08695	17 336.6
Moldova	1.10213	17 375.5
Sudan	1.06761	17 864.
Syria	1.08978	18 613.3
Fiji	1.09315	18 823.1
Armenia	1.09964	18 952.7
Zambia	1.08543	19 019.6
Iraq	1.08069	20 428.4
Philippines	1.08831	20 872.2
Sri Lanka	1.09217	21 291.5
Mauritania	1.05986	21 819.3
Guatemala	1.05738	24 008.9

Congo – Brazzaville	1.06629	24 261.6
Ghana	1.07235	24 598.8
Peru	1.07503	27 909.1
Nigeria	1.04228	28 962.6
Tajikistan	1.07556	32 155.1
Namibia	1.04037	35 995.4
Morocco	1.02061	36 211.1
China	1.04718	37 921.8
Maldives	1.0154	38 594.1
Angola	0.98797	38 825.4
Paraguay	1.03835	40 824.1
Jordan	1.05167	40 840.5
Panama	1.04885	41 873.3
Colombia	1.03659	41 889.3
Costa Rica	1.04088	42 168.8
Dominican Republic	1.03636	42 664.4
Indonesia	1.03462	43 353.1
Eswatini	0.991675	45 573.4
South Africa	1.03194	46 225.5
Ecuador	1.03659	47 489.9
Bulgaria	1.04661	48 222.2
Mongolia	1.03427	50 301.4
Kazakhstan	1.04071	54 155.4
Ukraine	1.03927	54 766.2
Albania	1.02889	55 332.6
Poland	1.0333	58 775.6
Jamaica	1.00868	58 870.
Tunisia	0.98585	61 766.2
Brazil	0.991098	66 127.5
Algeria	0.964614	66 251.
Mexico	0.996299	66 653.
Romania	1.01403	68 208.4
Chile	1.00795	69 223.4
Turkey	0.969502	69 519.3
Mauritius	0.983665	70 124.2
Argentina	1.00011	70 330.1

Botswana	0.993939	70 987.3
Thailand	0.984499	71 826.3
Uruguay	0.985189	72 188.9
Gabon	0.974677	74 159.2
Iran	0.952756	74 592.9
Malaysia	0.990722	77 318.1
Barbados	0.985732	78 050.3
Serbia	0.989064	85 112.5
Venezuela	0.957608	88 131.3
Lithuania	0.97558	94 687.4
Russia	0.971466	102 116.
Slovakia	0.982471	103 225.
Hungary	0.963823	106 932.
Malta	0.94932	107 797.
Croatia	0.953964	112 631.
New Zealand	0.95251	112 779.
Israel	0.960433	118 256.
Trinidad and Tobago	0.929153	122 342.
Estonia	0.946738	126 871.
Kuwait	0.853304	129 948.
Latvia	0.907326	139 490.
Saudi Arabia	0.868684	140 806.
South Korea	0.914693	153 299.
Slovenia	0.88944	171 724.
Taiwan	0.864406	176 683.
Canada	0.894342	180 419.
Czech Republic	0.88851	185 032.
United Kingdom	0.889037	187 914.
Bahrain	0.780323	188 486.
Greece	0.840642	188 775.
Australia	0.870029	191 471.
United States	0.879124	197 581.
Germany	0.869067	204 959.
Iceland	0.829663	205 845.
Cyprus	0.794408	213 757.
France	0.817322	217 624.

```
Spain                    0.792838   221542.
Finland                  0.832247   222340.
Portugal                 0.74826    223403.
Sweden                   0.831309   224000.
Japan                    0.840325   224738.
Italy                    0.799052   230750.
Denmark                  0.830046   231504.
Ireland                  0.799664   232858.
Austria                  0.816258   234837.
Macao                    0.753365   238788.
Netherlands              0.807095   243409.
Belgium                  0.778443   259390.
Norway                   0.7993     274993.
Hong Kong                0.767734   275231.
Switzerland              0.784399   297507.
Brunei                   0.704891   301131.
United Arab Emirates     0.702262   306399.
Singapore                0.725094   316631.
Qatar                    0.650793   359272.
Luxembourg               0.682636   386321.
```

Out[•]= ./ExploitTable2010.csv

In[•]:=
```
(* ExploitationArray=ArrayPlot[Transpose[ExploitationIndex],
  FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./ExploitationArray.eps",ExploitationArray,"EPS"] *)
```

In[•]:=
```
ExploitationIndexDistrib = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndexDistrib[[t, i]] = {t, ExploitationIndex[[t, i]]}
 ]
]
```

Below the Gini coefficient for the exploitation intensity index is plotted over T. The pattern of the Gini coefficient is consistent with the previous charts depicting the exploitation intensity index.

In[∘]:=
```
(* Gini=Table[0.0,{T}];
For[t=1,t≤T,t++,
 Gini[[t]]=N/(N-1) ((Σ_{i=1}^N Σ_{j=1}^N Abs[ExploitationIndex[[t,i]]-ExploitationIndex[[t,j]]])/(2 N^2 Mean[ExploitationIndex[[t]]]))
] *)
```

In[∘]:=
```
Clear[SortExploitationIndex];
SortExploitationIndex = Table[Table[0.0, {N}], {t, 1, T}];
GiniTest = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortExploitationIndex[[t]] = Sort[ExploitationIndex[[t]]];
 GiniTest[[t]] = N/(N - 1) Σ_{i=1}^N ((2 i - N - 1) SortExploitationIndex[[t, i]]) / (N^2 Mean[SortExploitationIndex[[t]]])
]
```

In[∘]:=
```
Chop[GiniTest]
```

Out[∘]=
```
{0.0671755}
```

In[∘]:=
```
(* ExploitationGini=ListLinePlot[GiniTest,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: e_t^y"}] *)
```

In[∘]:=
```
(* Export["./ExploitationGini.eps",ExploitationGini,"EPS"] *)
```

Plotting Gini coefficient for wealth $W_{t-1} = p_{t-1}\,\omega_{t-1}$.

```
In[•]:= Clear[SortWealth, WealthGini];
SortWealth = Table[Table[0.0, {N}], {t, 1, T}];
WealthGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortWealth[[t]] = If[t == 1, Sort[AgentSet[[All, 1]]], Sort[Activities[[t - 1, All, 5]]]];
 WealthGini[[t]] = N/(N - 1) Sum[(2 i - N - 1) SortWealth[[t, i]]/(N^2 Mean[SortWealth[[t]]]), {i, 1, N}]
]
(* WealthGiniPlot=
 ListLinePlot[WealthGini,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: W^γ_{t-1}"},LabelStyle→12,PlotRange→{0,1}] *)

In[•]:= (* Export["./WealthGini.eps",WealthGiniPlot,"EPS"] *)

In[•]:= WealthGini

Out[•]= {0.809975}
```

The figure below plots the distribution of wealth for select *t*.

```
In[•]:= (* WealthDistribRow=GraphicsRow[{
    Histogram[AgentSet[[All,1]],10,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 1",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[24,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 25",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[49,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 50",18],AxesLabel→{"ω^γ_{t-1}"}]
   },Spacings→{15,-20}
  ]
  Export["./WealthDistribRow.eps",WealthDistribRow,"EPS"]
*)
```

Income Figures

Figures on net and gross income.

In[•]:=
```
Clear[Income];
Income = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  Income⟦t, i⟧ = Profit⟦t⟧ × Activities⟦t, i, 5⟧ + Wage⟦t⟧ × Activities⟦t, i, 6⟧
 ]
]

IncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
```

$$\text{IncomeGini}[[t]] = \frac{N}{N-1}\left(\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\text{Abs}[\text{Income}[[t, i]] - \text{Income}[[t, j]]]\right) \Big/ \left(2\,N^2\,\text{Mean}[\text{Income}[[t]]]\right)\right)$$

```
]
(* IncomeGiniPlot=ListLinePlot[IncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[IncomeGini]+0.1},AxesLabel→{"t","Gini: Net Income"},LabelStyle→14] *)
```

In[•]:=
```
(* Export["./NetIncomeGini.eps",IncomeGiniPlot,"EPS"] *)
```

```
In[•]:= Clear[IncomeShares];
IncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  IncomeShares[[t, i]] = Income[[t, i]] / Total[Income[[t]]]
 ]
]
(* IncomeArray=ArrayPlot[Transpose[IncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},
  PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",ColorFunctionScaling→True,
  DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./NetIncomeArray.eps",IncomeArray,"EPS"] *)

In[•]:= Clear[GrossIncome];
GrossIncome = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncome[[t, i]] = (1 + Profit[[t]]) Activities[[t, i, 5]] + Wage[[t]] × Activities[[t, i, 6]]
 ]
]

GrossIncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 GrossIncomeGini[[t]] = N/(N - 1) ((Sum[Sum[Abs[GrossIncome[[t, i]] - GrossIncome[[t, j]]], {j, 1, N}], {i, 1, N}]) / (2 N^2 Mean[GrossIncome[[t]]]))
]
(* GrossIncomeGiniPlot=ListLinePlot[GrossIncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[GrossIncomeGini]+0.1},AxesLabel→{"t","Gini: Income"},LabelStyle→14] *)
```

```
In[•]:= (* Export["./GrossIncomeGini.eps",GrossIncomeGiniPlot,"EPS"] *)

In[•]:= GrossIncomeGini

Out[•]= {0.793268}

In[•]:= Clear[GrossIncomeShares];
GrossIncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncomeShares[[t, i]] = GrossIncome[[t, i]] / Total[GrossIncome[[t]]]
 ]
]
(* GrossIncomeArray=
 ArrayPlot[Transpose[GrossIncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./GrossIncomeArray.eps",GrossIncomeArray,"EPS"] *)
```

Updated Simulation Reporting

```
In[•]:= (ExploitedTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploitedTable2010.csv", ExploitedTable, "CSV"]

Out[•]//TableForm=
Burundi                  &   1.12857   \\
Mozambique               &   1.1257    \\
Mali                     &   1.12456   \\
Rwanda                   &   1.12625   \\
Malawi                   &   1.12622   \\
Liberia                  &   1.12594   \\
Congo - Kinshasa         &   1.12511   \\
```

```
Sierra Leone               &   1.12397   \\
Ethiopia                   &   1.12147   \\
Burkina Faso               &   1.11742   \\
Central African Republic   &   1.11753   \\
Niger                      &   1.1097    \\
Madagascar                 &   1.11719   \\
Myanmar                    &   1.11779   \\
Zimbabwe                   &   1.1219    \\
Togo                       &   1.11533   \\
Cambodia                   &   1.11416   \\
Ivory Coast                &   1.11083   \\
Uganda                     &   1.11638   \\
Kyrgyzstan                 &   1.12347   \\
Benin                      &   1.11048   \\
Kenya                      &   1.1167    \\
Nepal                      &   1.10851   \\
Pakistan                   &   1.11151   \\
Gambia                     &   1.10225   \\
Bangladesh                 &   1.10922   \\
Cameroon                   &   1.1087    \\
Yemen                      &   1.09325   \\
Senegal                    &   1.09116   \\
Tanzania                   &   1.09345   \\
Bolivia                    &   1.10967   \\
Lesotho                    &   1.10054   \\
Laos                       &   1.09406   \\
Vietnam                    &   1.10338   \\
Haiti                      &   1.08376   \\
El Salvador                &   1.09216   \\
Honduras                   &   1.09138   \\
Egypt                      &   1.09644   \\
Belize                     &   1.10607   \\
India                      &   1.08419   \\
Nicaragua                  &   1.08695   \\
Moldova                    &   1.10213   \\
Sudan                      &   1.06761   \\
```

```
Syria                    &   1.08978   \\
Fiji                     &   1.09315   \\
Armenia                  &   1.09964   \\
Zambia                   &   1.08543   \\
Iraq                     &   1.08069   \\
Philippines              &   1.08831   \\
Sri Lanka                &   1.09217   \\
Mauritania               &   1.05986   \\
Guatemala                &   1.05738   \\
Congo - Brazzaville      &   1.06629   \\
Ghana                    &   1.07235   \\
Peru                     &   1.07503   \\
Nigeria                  &   1.04228   \\
Tajikistan               &   1.07556   \\
Namibia                  &   1.04037   \\
Morocco                  &   1.02061   \\
China                    &   1.04718   \\
Maldives                 &   1.0154    \\
Paraguay                 &   1.03835   \\
Jordan                   &   1.05167   \\
Panama                   &   1.04885   \\
Colombia                 &   1.03659   \\
Costa Rica               &   1.04088   \\
Dominican Republic       &   1.03636   \\
Indonesia                &   1.03462   \\
South Africa             &   1.03194   \\
Ecuador                  &   1.03659   \\
Bulgaria                 &   1.04661   \\
Mongolia                 &   1.03427   \\
Kazakhstan               &   1.04071   \\
Ukraine                  &   1.03927   \\
Albania                  &   1.02889   \\
Poland                   &   1.0333    \\
Jamaica                  &   1.00868   \\
Romania                  &   1.01403   \\
Chile                    &   1.00795   \\
Argentina                &   1.00011   \\
```

Out[•]= ./ExploitedTable2010.csv

In[•]:=
```
(ExploiterTable = DeleteCases[Table[If[ExploitationIndex⟦1, i⟧ < 1.0,
      {CountryList⟦i⟧, "&", ExploitationIndex⟦1, i⟧, "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploiterTable2010.csv", ExploiterTable, "CSV"]
```

Out[•]//TableForm=

Angola	&	0.98797	\\
Eswatini	&	0.991675	\\
Tunisia	&	0.98585	\\
Brazil	&	0.991098	\\
Algeria	&	0.964614	\\
Mexico	&	0.996299	\\
Turkey	&	0.969502	\\
Mauritius	&	0.983665	\\
Botswana	&	0.993939	\\
Thailand	&	0.984499	\\
Uruguay	&	0.985189	\\
Gabon	&	0.974677	\\
Iran	&	0.952756	\\
Malaysia	&	0.990722	\\
Barbados	&	0.985732	\\
Serbia	&	0.989064	\\
Venezuela	&	0.957608	\\
Lithuania	&	0.97558	\\
Russia	&	0.971466	\\
Slovakia	&	0.982471	\\
Hungary	&	0.963823	\\
Malta	&	0.94932	\\
Croatia	&	0.953964	\\
New Zealand	&	0.95251	\\
Israel	&	0.960433	\\
Trinidad and Tobago	&	0.929153	\\
Estonia	&	0.946738	\\
Kuwait	&	0.853304	\\
Latvia	&	0.907326	\\
Saudi Arabia	&	0.868684	\\

```
South Korea            &   0.914693   \\
Slovenia               &   0.88944    \\
Taiwan                 &   0.864406   \\
Canada                 &   0.894342   \\
Czech Republic         &   0.88851    \\
United Kingdom         &   0.889037   \\
Bahrain                &   0.780323   \\
Greece                 &   0.840642   \\
Australia              &   0.870029   \\
United States          &   0.879124   \\
Germany                &   0.869067   \\
Iceland                &   0.829663   \\
Cyprus                 &   0.794408   \\
France                 &   0.817322   \\
Spain                  &   0.792838   \\
Finland                &   0.832247   \\
Portugal               &   0.74826    \\
Sweden                 &   0.831309   \\
Japan                  &   0.840325   \\
Italy                  &   0.799052   \\
Denmark                &   0.830046   \\
Ireland                &   0.799664   \\
Austria                &   0.816258   \\
Macao                  &   0.753365   \\
Netherlands            &   0.807095   \\
Belgium                &   0.778443   \\
Norway                 &   0.7993     \\
Hong Kong              &   0.767734   \\
Switzerland            &   0.784399   \\
Brunei                 &   0.704891   \\
United Arab Emirates   &   0.702262   \\
Singapore              &   0.725094   \\
Qatar                  &   0.650793   \\
Luxembourg             &   0.682636   \\
```

Out[•]= ./ExploiterTable2010.csv

In[•]:=
```
(ExploitIncomeCSV = Table[{CountryList⟦i⟧, "&", ExploitationIndex⟦1, i⟧, "\\\\"}, {i, 1, N}]) // TableForm
Export["./ExploitIncomeTable2010.csv", ExploitIncomeCSV, "CSV"]
```

Out[•]//TableForm=

Burundi	&	1.12857	\\
Mozambique	&	1.1257	\\
Mali	&	1.12456	\\
Rwanda	&	1.12625	\\
Malawi	&	1.12622	\\
Liberia	&	1.12594	\\
Congo - Kinshasa	&	1.12511	\\
Sierra Leone	&	1.12397	\\
Ethiopia	&	1.12147	\\
Burkina Faso	&	1.11742	\\
Central African Republic	&	1.11753	\\
Niger	&	1.1097	\\
Madagascar	&	1.11719	\\
Myanmar	&	1.11779	\\
Zimbabwe	&	1.1219	\\
Togo	&	1.11533	\\
Cambodia	&	1.11416	\\
Ivory Coast	&	1.11083	\\
Uganda	&	1.11638	\\
Kyrgyzstan	&	1.12347	\\
Benin	&	1.11048	\\
Kenya	&	1.1167	\\
Nepal	&	1.10851	\\
Pakistan	&	1.11151	\\
Gambia	&	1.10225	\\
Bangladesh	&	1.10922	\\
Cameroon	&	1.1087	\\
Yemen	&	1.09325	\\
Senegal	&	1.09116	\\
Tanzania	&	1.09345	\\
Bolivia	&	1.10967	\\
Lesotho	&	1.10054	\\
Laos	&	1.09406	\\

```
Vietnam                    &   1.10338    \\
Haiti                      &   1.08376    \\
El Salvador                &   1.09216    \\
Honduras                   &   1.09138    \\
Egypt                      &   1.09644    \\
Belize                     &   1.10607    \\
India                      &   1.08419    \\
Nicaragua                  &   1.08695    \\
Moldova                    &   1.10213    \\
Sudan                      &   1.06761    \\
Syria                      &   1.08978    \\
Fiji                       &   1.09315    \\
Armenia                    &   1.09964    \\
Zambia                     &   1.08543    \\
Iraq                       &   1.08069    \\
Philippines                &   1.08831    \\
Sri Lanka                  &   1.09217    \\
Mauritania                 &   1.05986    \\
Guatemala                  &   1.05738    \\
Congo - Brazzaville        &   1.06629    \\
Ghana                      &   1.07235    \\
Peru                       &   1.07503    \\
Nigeria                    &   1.04228    \\
Tajikistan                 &   1.07556    \\
Namibia                    &   1.04037    \\
Morocco                    &   1.02061    \\
China                      &   1.04718    \\
Maldives                   &   1.0154     \\
Angola                     &   0.98797    \\
Paraguay                   &   1.03835    \\
Jordan                     &   1.05167    \\
Panama                     &   1.04885    \\
Colombia                   &   1.03659    \\
Costa Rica                 &   1.04088    \\
Dominican Republic         &   1.03636    \\
Indonesia                  &   1.03462    \\
```

```
Eswatini                    &    0.991675    \\
South Africa                &    1.03194     \\
Ecuador                     &    1.03659     \\
Bulgaria                    &    1.04661     \\
Mongolia                    &    1.03427     \\
Kazakhstan                  &    1.04071     \\
Ukraine                     &    1.03927     \\
Albania                     &    1.02889     \\
Poland                      &    1.0333      \\
Jamaica                     &    1.00868     \\
Tunisia                     &    0.98585     \\
Brazil                      &    0.991098    \\
Algeria                     &    0.964614    \\
Mexico                      &    0.996299    \\
Romania                     &    1.01403     \\
Chile                       &    1.00795     \\
Turkey                      &    0.969502    \\
Mauritius                   &    0.983665    \\
Argentina                   &    1.00011     \\
Botswana                    &    0.993939    \\
Thailand                    &    0.984499    \\
Uruguay                     &    0.985189    \\
Gabon                       &    0.974677    \\
Iran                        &    0.952756    \\
Malaysia                    &    0.990722    \\
Barbados                    &    0.985732    \\
Serbia                      &    0.989064    \\
Venezuela                   &    0.957608    \\
Lithuania                   &    0.97558     \\
Russia                      &    0.971466    \\
Slovakia                    &    0.982471    \\
Hungary                     &    0.963823    \\
Malta                       &    0.94932     \\
Croatia                     &    0.953964    \\
New Zealand                 &    0.95251     \\
Israel                      &    0.960433    \\
```

```
Trinidad and Tobago        &   0.929153   \\
Estonia                    &   0.946738   \\
Kuwait                     &   0.853304   \\
Latvia                     &   0.907326   \\
Saudi Arabia               &   0.868684   \\
South Korea                &   0.914693   \\
Slovenia                   &   0.88944    \\
Taiwan                     &   0.864406   \\
Canada                     &   0.894342   \\
Czech Republic             &   0.88851    \\
United Kingdom             &   0.889037   \\
Bahrain                    &   0.780323   \\
Greece                     &   0.840642   \\
Australia                  &   0.870029   \\
United States              &   0.879124   \\
Germany                    &   0.869067   \\
Iceland                    &   0.829663   \\
Cyprus                     &   0.794408   \\
France                     &   0.817322   \\
Spain                      &   0.792838   \\
Finland                    &   0.832247   \\
Portugal                   &   0.74826    \\
Sweden                     &   0.831309   \\
Japan                      &   0.840325   \\
Italy                      &   0.799052   \\
Denmark                    &   0.830046   \\
Ireland                    &   0.799664   \\
Austria                    &   0.816258   \\
Macao                      &   0.753365   \\
Netherlands                &   0.807095   \\
Belgium                    &   0.778443   \\
Norway                     &   0.7993     \\
Hong Kong                  &   0.767734   \\
Switzerland                &   0.784399   \\
Brunei                     &   0.704891   \\
United Arab Emirates       &   0.702262   \\
```

```
Singapore                    &    0.725094    \\
Qatar                        &    0.650793    \\
Luxembourg                   &    0.682636    \\
```

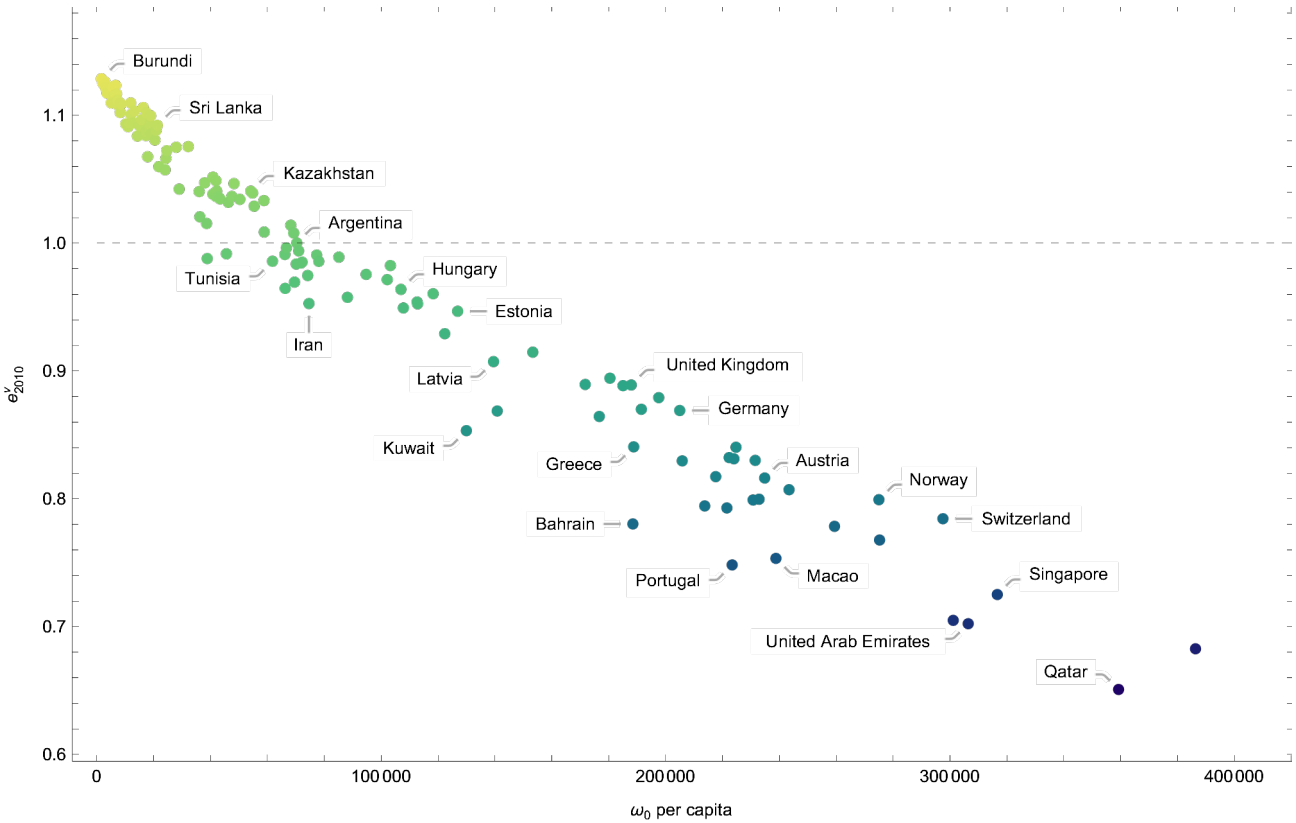
Out[•]= ./ExploitIncomeTable2010.csv

In[•]:= **DataNoHeader[[1]]**

Out[•]= {Burundi, 14 188.5, 1.298, 8.6756, 1635.45}

In[∘]:=
```
ExploitWealthPlot = ListPlot[Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, 1, N}] → CountryList,
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_2010"}, ColorFunction → "BlueGreenYellow",
  Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitWealthPlot2010.eps", ExploitWealthPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploitWealthPlot2010.eps

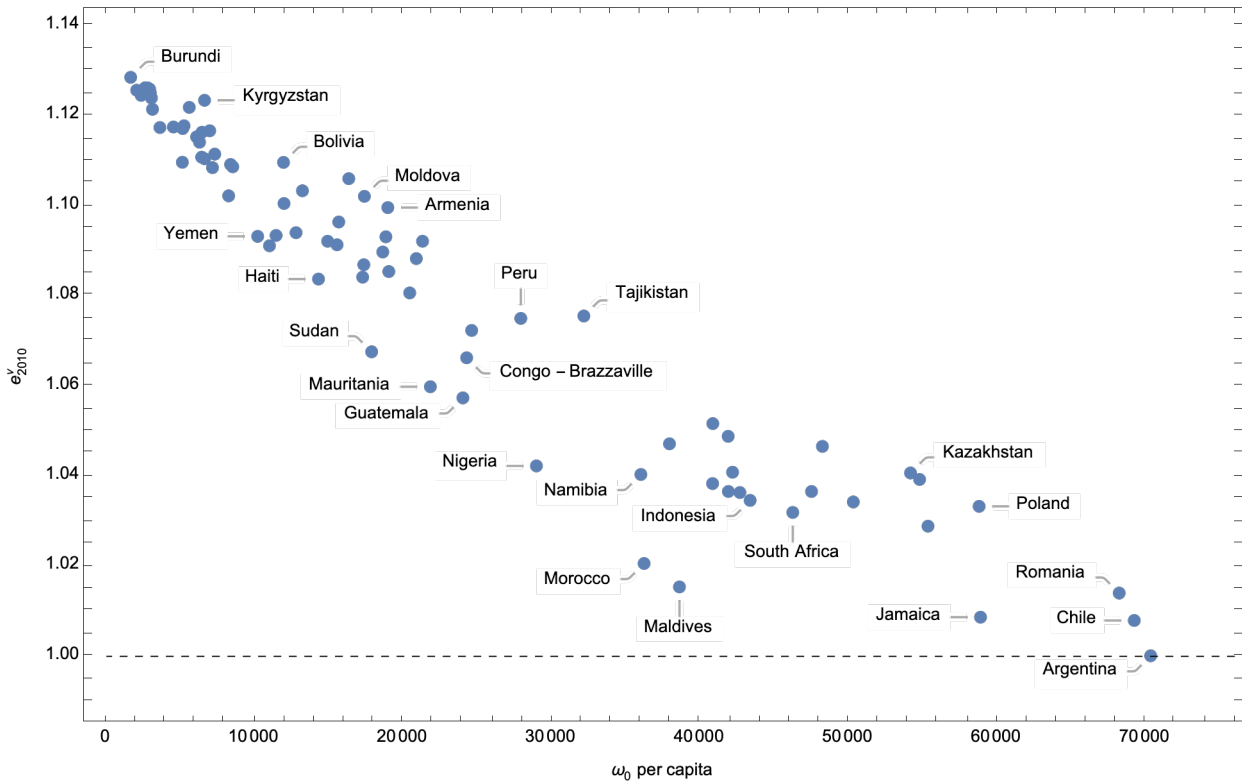
In[∘]:=
```
ExploitedCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1, i, 0.], {i, 1, N}], 0.]
ExploiterCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1, i, 0.], {i, 1, N}], 0.]
```

Out[∘]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28,
 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54,
 55, 56, 57, 58, 59, 60, 61, 63, 64, 65, 66, 67, 68, 69, 71, 72, 73, 74, 75, 76, 77, 78, 79, 84, 85, 88}
```

Out[∘]=
```
{62, 70, 80, 81, 82, 83, 86, 87, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104,
 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124,
 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144}
```

In[∘]:=
```
ExploitedCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploitedCountries}] → CountryList[[ExploitedCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^y_2010"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitedCountriesPlot2010.eps", ExploitedCountriesPlot, "EPS"]
```

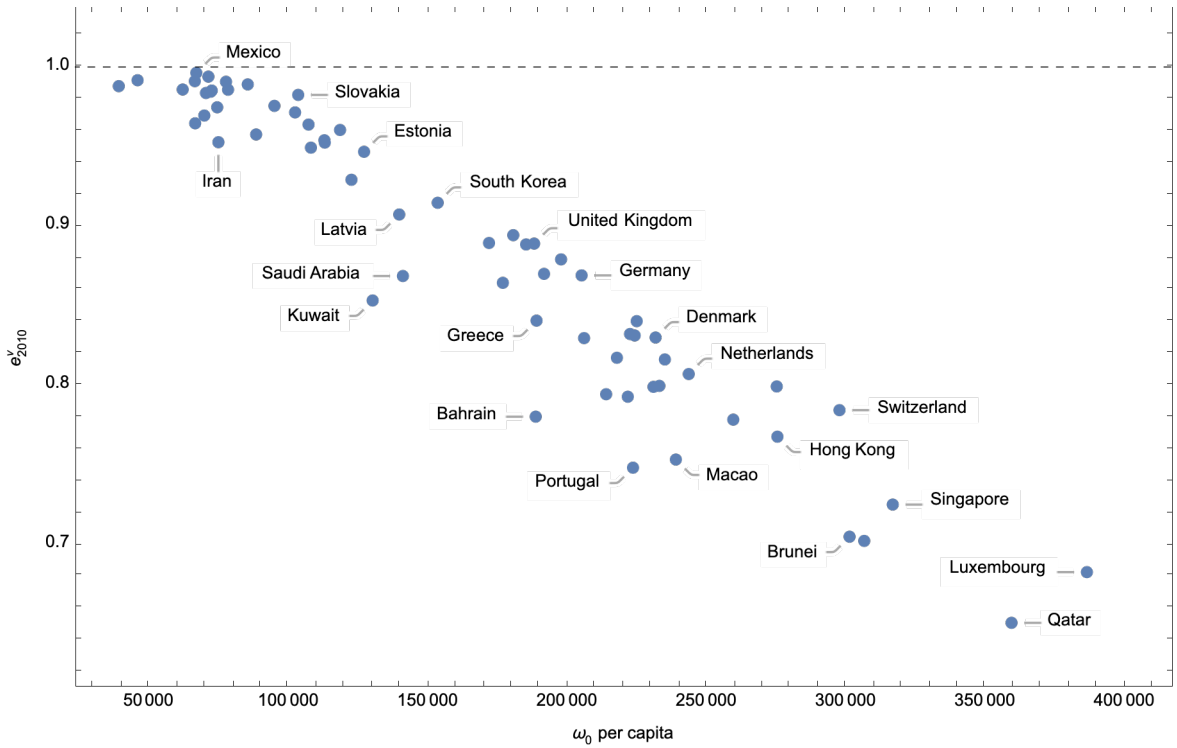
Out[∘]=



Out[∘]= ./ExploitedCountriesPlot2010.eps

In[∘]:=
```
ExploiterCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploiterCountries}] → CountryList[[ExploiterCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_2010"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploiterCountriesPlot2010.eps", ExploiterCountriesPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploiterCountriesPlot2010.eps

In[∘]:=
```
(outlierTesting = Table[{CountryList[[i]], AgentSet[[i, 1]], AgentSet[[i, 2]]}, {i, 1, Length[CountryList]}]) // TableForm
Export["./outlierTesting2010.csv", outlierTesting, "CSV"];
```

Out[∘]//TableForm=

Burundi	14 188.5	1.12609×10^6
Mozambique	48 024.4	2.77885×10^6

Mali	35 220.3	1.83994×10^6
Rwanda	26 239.	1.60137×10^6
Malawi	40 432.3	2.45995×10^6
Liberia	11 328.1	671 111.
Congo – Kinshasa	191 139.	1.04768×10^7
Sierra Leone	19 438.2	966 638.
Ethiopia	272 075.	1.12311×10^7
Burkina Faso	56 276.9	1.81972×10^6
Central African Republic	19 797.	644 114.
Niger	84 230.9	1.92184×10^6
Madagascar	108 697.	3.47267×10^6
Myanmar	264 555.	8.73066×10^6
Zimbabwe	70 862.3	3.01266×10^6
Togo	39 014.	1.13433×10^6
Cambodia	89 775.9	2.46986×10^6
Ivory Coast	131 729.	3.14216×10^6
Uganda	208 796.	6.39517×10^6
Kyrgyzstan	35 843.7	1.71177×10^6
Benin	60 880.2	1.43169×10^6
Kenya	292 414	9.10343×10^6
Nepal	193 039.	4.21163×10^6
Pakistan	1 309 276	3.21026×10^7
Gambia	14 764.9	261 502.
Bangladesh	1.23424×10^6	2.76468×10^7
Cameroon	173 103.	3.8036×10^6
Yemen	235 900.	3.27943×10^6
Senegal	139 264.	1.84289×10^6
Tanzania	492 417.	6.87873×10^6
Bolivia	119 857.	2.73191×10^6
Lesotho	23 870.4	402 073.
Laos	79 813.2	1.13173×10^6
Vietnam	1 161 342	2.12952×10^7

Haiti	142 101.	1.60512×10^6
El Salvador	92 151.8	1.24828×10^6
Honduras	129 203.	1.7184×10^6
Egypt	1.29516×10^6	1.95002×10^7
Belize	5264.35	105 358.
India	21 301 734	2.42672×10^8
Nicaragua	100 970.	1.2177×10^6
Moldova	70 998.1	1.25317×10^6
Sudan	617 112.	5.2536×10^6
Syria	397 628.	5.09923×10^6
Fiji	16 184.1	224 468.
Armenia	54 532.5	895 186.
Zambia	258 781.	3.02339×10^6
Iraq	607 581.	6.46561×10^6
Philippines	1.96129×10^6	2.43506×10^7
Sri Lanka	431 402.	5.84469×10^6
Mauritania	76 241.	579 024.
Guatemala	351 260.	2.57788×10^6
Congo – Brazzaville	103 687.	864 911.
Ghana	609 548.	5.59821×10^6
Peru	810 138.	7.78494×10^6
Nigeria	4.59067×10^6	2.78791×10^7
Tajikistan	242 044.	2.34742×10^6
Namibia	76 270.6	453 169.
Morocco	1.17119×10^6	5.66107×10^6
China	51 907 732	3.34127×10^8
Maldives	14 113.9	64 952.
Angola	906 813.	3.30981×10^6
Paraguay	255 069.	1.4814×10^6
Jordan	296 563.	2.01877×10^6
Panama	152 532.	1.00214×10^6
Colombia	1 894 346	1.07901×10^7

Costa Rica	193 023.	1.1535×10^6
Dominican Republic	413 635.	2.35009×10^6
Indonesia	10 484 265	5.84465×10^7
Eswatini	48 526.6	182 336.
South Africa	2.36753×10^6	1.28268×10^7
Ecuador	712 876.	4.0605×10^6
Bulgaria	358 050.	2.28905×10^6
Mongolia	136 815.	759 831.
Kazakhstan	880 149.	5.24966×10^6
Ukraine	2.50786×10^6	1.47162×10^7
Albania	163 120.	856 129.
Poland	2.25286×10^6	1.23821×10^7
Jamaica	165 454.	716 565.
Tunisia	656 896.	2.35878×10^6
Brazil	12 942 057	4.84078×10^7
Algeria	2.38355×10^6	7.33905×10^6
Mexico	7.60464×10^6	2.96551×10^7
Romania	1.39636×10^6	6.34567×10^6
Chile	1.18112×10^6	5.08258×10^6
Turkey	5 028 124	1.60161×10^7
Mauritius	87 515.	309 067.
Argentina	2.87621×10^6	1.15739×10^7
Botswana	141 059.	539 676.
Thailand	4 826 366	1.71529×10^7
Uruguay	242 504.	866 397.
Gabon	120 442.	398 034.
Iran	5 502 156	1.56524×10^7
Malaysia	2.18099×10^6	8.13349×10^6
Barbados	22 018.	78 990.8
Serbia	620 589.	2.28447×10^6
Venezuela	2.50644×10^6	7.3611×10^6
Lithuania	295 784.	983 903.
Russia	14 651 546	4.73209×10^7

Slovakia	557 857.	1.95247×10^6
Hungary	1.06156×10^6	3.25093×10^6
Malta	44 660.2	124 265.
Croatia	487 488.	1.39775×10^6
New Zealand	492 855.	1.39983×10^6
Israel	868 753.	2.60004×10^6
Trinidad and Tobago	162 482.	399 426.
Estonia	169 005.	462 585.
Kuwait	388 792.	642 540.
Latvia	295 564.	641 900.
Saudi Arabia	3.86112×10^6	6.86936×10^6
South Korea	7 595 314	1.71819×10^7
Slovenia	350 883.	692 903.
Taiwan	4.0886×10^6	7.12439×10^6
Canada	6 160 862	1.24809×10^7
Czech Republic	1.94959×10^6	3.83149×10^6
United Kingdom	11 924 992	2.34998×10^7
Bahrain	233 892.	281 027.
Greece	2.0553×10^6	3.20302×10^6
Australia	4 241 993	7.59685×10^6
United States	61 054 648	1.1439×10^8
Germany	16 566 225	2.95285×10^7
Iceland	65 932.1	97 778.
Cyprus	177 290.	225 796.
France	14 135 247	1.98521×10^7
Spain	10 397 201	1.31552×10^7
Finland	1.19303×10^6	1.78987×10^6
Portugal	2.3672×10^6	2.50206×10^6
Sweden	2.10341×10^6	3.14243×10^6
Japan	28 888 310	4.49551×10^7
Italy	13 689 293	1.7778×10^7
Denmark	1.28596×10^6	1.91035×10^6

Ireland	1.06051×10^6	1.38082×10^6
Austria	1.97496×10^6	2.76089×10^6
Macao	128 516.	138 587.
Netherlands	4.06076×10^6	5.45681×10^6
Belgium	2.83739×10^6	3.38312×10^6
Norway	1.34359×10^6	1.74671×10^6
Hong Kong	1 917 344	2.18916×10^6
Switzerland	2.32314×10^6	2.83838×10^6
Brunei	117 019.	104 805.
United Arab Emirates	2 619 715	2.32338×10^6
Singapore	1 624 696	1.57035×10^6
Qatar	666 917.	490 026.
Luxembourg	196 212.	161 837.

2019

2019 - Model Setup

2019 - Basic Model

In[ ]:=
```
Clear[Activities, i, Wage, Profit, u];
(Activities = Table[Table[{0.0, 0.0, 0.0, 0.0, 0.0, 0.0}, {N}], {T}]);
Wage = Table[0.0, {T}];
Profit = Table[0.0, {T}];

(* First Stage t=1 simulation *)
(* Setting w1 and π1 *)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
```

```
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]], Wage[[1]] = b, Wage[[1]] = RandomReal[{b, (1 - A (1 + 0))/L}]],
  Wage[[1]] = (1 - A (1 + 0))/L];
Profit[[1]] = ((p - p0 A - Wage[[1]] L)/(p0 A)) /. {p → 1, p0 → 1};

(* Checking whether the simulation is capital constrained, labour constrained,
or on the knife-edge and calling the corresponding function to find optimal (x_t^ν, y_t^ν, z_t^ν, δ_t^ν) for all ν and t=1*)
If[Total[l] ≥ L A^-1 Total[AgentSet[[All, 1]]],
  If[Total[l] > L A^-1 Total[AgentSet[[All, 1]]],
   CapitalConstrained[1, A, L, l, 1, Profit[[1]], b], KnifeEdge[1, A, L, l, 1, Profit[[1]], b]],
  LabourConstrained[1, A, L, l, 1, Profit[[1]], b]
 ];

(* Simulation for remaining T-1 time steps *)
For[t = 2, t ≤ T, t++,

 (* Updating w_t *)
 If[Total[l] ≥ L A^-1 Total[Activities[[t - 1, All, 5]]], If[Total[l] > L A^-1 Total[Activities[[t - 1, All, 5]]],
   Wage[[t]] = b, Wage[[t]] = RandomReal[{b, ((1 - A (1 + r))/L /. r → 0)}]], Wage[[t]] = ((1 - A (1 + r))/L /. r → 0)];

 (* Updating r_t *)
 Profit[[t]] = ((p - p0 A - Wage[[t]] L)/(p0 A)) /. {p → 1, p0 → 1};

 (* Checking whether the simulation is capital constrained, labour constrained,
 or on the knife-edge and calling the corresponding function to find optimal (x_t^ν, y_t^ν, z_t^ν, δ_t^ν) for all ν *)
 If[Total[l] ≥ L A^-1 Total[Activities[[t - 1, All, 5]]],
```

```
  If[Total[l] > L A^-1 Total[Activities⟦t - 1, All, 5⟧],
   CapitalConstrained[1, A, L, l, t, Profit⟦t⟧, b], KnifeEdge[1, A, L, l, t, Profit⟦t⟧, b]],
  LabourConstrained[1, A, L, l, t, Profit⟦t⟧, b]
 ];

]
```

Exploitation and Class Over Time

The chart below captures the number of agents who are exploiters, exploited, or neither over the course of the simulation according to Definition 2, which defines exploitation in relation to Λ_t^v and $v_t\, c_t^v$, where v_t is just the embodied labour value and $c_t^v = \pi_t\, W_{t-1}^v + (w_t - b_t)\; l^v + b_t\, \Lambda_t^v$. An agent v is exploited during t if and only if $\Lambda_t^v > v_t\, c_t^v$, an agent is an exploiter if and only if $\Lambda_t^v < v_t\, c_t^v$, and an agent is neither exploited nor an exploiter if and only if $\Lambda_t^v = v_t\, c_t^v$.

In[ ]:=
```
Clear[IndivExploitation, ConsumptionBundle];
ConsumptionBundle = Table[0.0, {i, T}, {j, N}];
IndivExploitation = Table[0.0, {i, T}, {j, N}];

For[i = 1, i ≤ N, i++,
 ConsumptionBundle⟦1, i⟧ = EmbodiedValues⟦1, 2⟧ (Profit⟦1⟧ × AgentSet⟦i, 1⟧ + (Wage⟦1⟧ - b) l⟦i⟧ + b Activities⟦1, i, 6⟧);
 IndivExploitation⟦1, i⟧ = Activities⟦1, i, 6⟧ - ConsumptionBundle⟦1, i⟧;

]

For[t = 2, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ConsumptionBundle⟦t, i⟧ =
   EmbodiedValues⟦1, 2⟧ (Profit⟦t⟧ × Activities⟦t - 1, i, 5⟧ + (Wage⟦t⟧ - b) l⟦i⟧ + b Activities⟦t, i, 6⟧);
  IndivExploitation⟦t, i⟧ = Activities⟦t, i, 6⟧ - ConsumptionBundle⟦t, i⟧;
 ]
]
```

```
Clear[Exploiters, Exploited, NonExploit];
Exploiters = Table[0.0, {T}];
Exploited = Table[0.0, {T}];
NonExploit = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  If[IndivExploitation⟦t, i⟧ < 0, Exploiters⟦t⟧ ++, If[IndivExploitation⟦t, i⟧ == 0, NonExploit⟦t⟧ ++, Exploited⟦t⟧ ++]];
 ]
]

ExploitationLegend =
  Grid[{{Row[{Graphics[{Thick, Blue, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploiters", 20], FontFamily → "Times"]}], Spacer[10],
      Row[{Graphics[{Thick, Red, Dotted, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Neither Exploiter or Exploited", 20], FontFamily → "Times"]}]},
     {Row[{Graphics[{Thick, Black, Dashed, Line[{{0, 0}, {1, 0}}]}, AspectRatio → 0.15, ImageSize → Scaled[0.04]],
        Spacer[5], Style[Style["Exploited", 20], FontFamily → "Times"]}], Spacer[10],
     }}, Frame → True, Alignment → Left];
(* ExploitationPlot=Labeled[
  ListLinePlot[{Exploiters,Exploited,NonExploit},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black},{Thick,Dotted,Red}},
   Frame→True,FrameLabel→{"t","Total ν in Group"},LabelStyle→20,
   PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],ExploitationLegend] *)
```

In[∘]:=
```
Export["./ExploitationPlot.eps", ExploitationPlot, "EPS"]
```

Out[∘]=
```
./ExploitationPlot.eps
```

The chart below captures the class composition of the simulation over time according to Corollary 1 of Theorem 3 (class is defined using labour endowments l^{ν} in relation to means of production).

In[∘]:=
```
Clear[Class1, Class2, Class3, Class4, j, k];
Class1 = Table[0.0, {T}];
Class2 = Table[0.0, {T}];
```

```
Class3 = Table[0.0, {T}];
Class4 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ == Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class2⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && Profit⟦j⟧ > 0, Class3⟦j⟧++, 0];
 ]
]

Clear[k];
For[k = 1, k ≤ N, k++,
 If[AgentSet⟦k, 1⟧ == 0 && Profit⟦1⟧ > 0, Class4⟦1⟧++, 0];
]

Clear[j, k];
For[j = 2, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
```

```
  If[Activities⟦j - 1, k, 5⟧ == 0 && Profit⟦j⟧ > 0, Class4⟦j⟧ ++, 0];
 ]
]


(* ClassLegend=
 Column[{
   Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^1 = {∑v ∈ (+,0,+)\(+,0,0); A_t y_t^v < z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,DotDashed,Green,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^2 = {∑v ∈ (+,0,0); A_t y_t^v = z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Purple,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^3 = {∑v ∈ (+,+,0)\(+,0,0); A_t y_t^v > z_t^v}",20],FontFamily→"Times"]}],
   Row[{Graphics[{Thick,Black,Dotted,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
     Spacer[5],Style[Style["C_t^4 = {∑v ∈ (0,+,0); W_{t-1}^v = 0}",20],FontFamily→"Times"]}]
  },Frame→True,Alignment→Left];
ClassPlotCorollary1=Labeled[
  ListLinePlot[{Class1,Class2,Class3,Class4},PlotStyle→{{Thick,Blue},{Thick,DotDashed,Green},
     {Thick,Dashed,Purple},{Thick, Dotted,Black},{Thick,DotDashed,Orange},{Thick,Dashed,Green}},
   PlotRange→{{1,T},{-10,N+10}},Frame→True,FrameLabel→{"t","Total v in Classes"},
   LabelStyle→20,ImageSize→{500,350},AxesOrigin→{1,-10}],ClassLegend] *)
```

In[∘]:=
```
(* Export["./ClassPlotCorollary1.eps",ClassPlotCorollary1,"EPS"] *)
```

The chart below captures the intersection of class and exploitation over the simulation. If an agent is in C^1 according to Corollary 1 of Theorem 3 and is also an exploiter according to Definition 2, then the agent is in the group "$C^1 \wedge$ Exploiter". If an agent is in $C^3 \cup C^4$ and is exploited they are in group "$(C^3 \cup C^4) \wedge$ Exploited".

In[∘]:=
```
Clear[CECP1, CECP2, CECP3, j, k];
CECP1 = Table[0.0, {T}];
CECP2 = Table[0.0, {T}];
CECP3a = Table[0.0, {T}];
```

```
CECP3b = Table[0.0, {T}];
CECP3 = Table[0.0, {T}];

For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ < Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ < 0, CECP1⟦j⟧++, 0];
 ]
]

Clear[j, k];
For[j = 1, j ≤ T, j++,
 For[k = 1, k ≤ N, k++,
  If[A Activities⟦j, k, 2⟧ > Activities⟦j, k, 3⟧ && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
  If[If[j == 1, AgentSet⟦k, 1⟧, Activities⟦j - 1, k, 5⟧] == 0 && IndivExploitation⟦j, k⟧ > 0, CECP3⟦j⟧++, 0];
 ]
]


(* CECPLegend=
 Grid[{{Row[{Graphics[{Thick,Blue,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],Spacer[5],
      Style[Style["∑v ∈ C_t^1 ∧ Exploiter",20],FontFamily→"Times"]}],Spacer[10],
    Row[{Graphics[{Thick,Black,Dashed,Line[{{0,0},{1,0}}]},AspectRatio→0.15,ImageSize→Scaled[0.04]],
      Spacer[5],Style[Style["∑v ∈ (C_t^3 ⋃ C_t^4) ∧ Exploited",20],FontFamily→"Times"]}],
   }},Frame→True,Alignment→Left];
CECPPlotCorollary1=
 Labeled[ListLinePlot[{CECP1,CECP3},PlotStyle→{{Thick,Blue},{Thick,Dashed,Black}},Frame→True,FrameLabel→
    {"t","Total v in Group"},LabelStyle→20,PlotRange→{{1,T},{-10,N+10}},ImageSize→{500,350}],CECPLegend] *)
```

In[∘]:=
```
(* Export["./CECPPlotCorollary1.eps",CECPPlotCorollary1,"EPS"] *)
```

The charts below display the distribution of an index of the intensity of exploitation across the agents over the simulation. The index itself is calculated as $e_t^v = \Lambda_t^v / v_t\, c_t^v$ and the charts that follow explore some possibilities for visualizing the intensity of exploitation over time.

In[∘]:=
```
Clear[ExploitationIndex];
ExploitationIndex = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndex[[t, i]] = Activities[[t, i, 6]] / (EmbodiedValues[[1, 2]] ConsumptionBundle[[t, i]] / EmbodiedValues[[1, 2]]) /. p → 1
 ]
]
```

The chart below uses *Mathematica's* `ArrayPlot` function to display the exploitation index of the N agents over *T*. There is a fixed distribution of uneven exploitation intensity while the simulation is capital constrained, however, this pattern disappears once the simulation is labour constrained, at which point exploitation intensity is the same for all $v \in \mathcal{N}$.

In[∘]:=
```
ExploitationIndex[[1, 1]]
```

Out[∘]=
```
1.13295
```

In[∘]:=
```
ExploitationIndex[[1]]
```

Out[∘]=
```
{1.13295, 1.13023, 1.12934, 1.1265, 1.12224, 1.12473, 1.11991, 1.1252, 1.12737, 1.11894, 1.12706, 1.1218,
 1.12026, 1.11247, 1.11545, 1.11906, 1.11382, 1.11477, 1.11224, 1.11592, 1.12239, 1.11286, 1.10972,
 1.10586, 1.10442, 1.11338, 1.10199, 1.10259, 1.09634, 1.11613, 1.09498, 1.10775, 1.09377, 1.0947, 1.10345,
 1.0936, 1.10117, 1.10353, 1.08421, 1.09016, 1.09878, 1.07643, 1.08947, 1.08558, 1.08825, 1.09978, 1.07019,
 1.07599, 1.07146, 1.06728, 1.07183, 1.09498, 1.07275, 1.07969, 1.0701, 1.06501, 1.07611, 1.08197, 1.07412,
 1.07251, 1.06797, 1.0594, 1.03096, 0.999419, 1.05311, 1.05717, 1.05043, 1.052, 1.03913, 1.04893, 1.04675,
 1.01988, 1.03602, 1.03889, 1.03419, 1.03603, 1.0199, 0.998288, 1.0113, 1.01205, 1.01944, 0.999177,
 1.00849, 1.00042, 1.00586, 0.984658, 1.02076, 0.992077, 1.01416, 0.982677, 1.00184, 0.976184, 0.944714,
 0.969257, 0.950517, 0.9239, 0.94572, 0.960804, 0.955007, 0.949787, 0.9382, 0.967598, 0.967681, 0.954984,
 0.944326, 0.923317, 0.936674, 0.906679, 0.894906, 0.835889, 0.899167, 0.892801, 0.884313, 0.885706,
 0.76948, 0.866503, 0.877336, 0.866954, 0.865302, 0.835101, 0.830534, 0.81157, 0.860228, 0.811982,
 0.732118, 0.81773, 0.826464, 0.834457, 0.81348, 0.820439, 0.740576, 0.781844, 0.775851, 0.804695,
 0.78501, 0.739533, 0.758752, 0.811127, 0.768917, 0.725552, 0.717684, 0.634108, 0.63912, 0.620679}
```

In[∘]:=
```
CountryList
```

Out[∘]=
```
{Malawi, Burundi, Sierra Leone, Congo - Kinshasa, Mali, Madagascar, Mozambique, Rwanda, Uganda, Burkina Faso, Zimbabwe,
 Liberia, Gambia, Niger, Ethiopia, Togo, Central African Republic, Tanzania, Sudan, Benin, Venezuela, Pakistan,
 Cameroon, Ivory Coast, Senegal, Kenya, Nepal, Cambodia, Lesotho, Kyrgyzstan, Haiti, Ghana, Myanmar, Nigeria,
 Zambia, Bangladesh, Egypt, Bolivia, Yemen, Nicaragua, Vietnam, Mauritania, Syria, Honduras, Philippines, Belize,
 Guatemala, India, Congo - Brazzaville, Laos, Eswatini, Moldova, Iraq, Tunisia, El Salvador, Namibia, Fiji,
 Armenia, Jordan, Tajikistan, Sri Lanka, Paraguay, Morocco, Angola, Colombia, Peru, Costa Rica, South Africa,
 Jamaica, Mongolia, Trinidad and Tobago, Algeria, Gabon, Kazakhstan, Brazil, Bulgaria, Ecuador, Indonesia,
 China, Dominican Republic, Argentina, Mauritius, Albania, Thailand, Botswana, Iran, Poland, Mexico, Serbia,
 Uruguay, Romania, Chile, Maldives, Malaysia, Barbados, Turkey, Panama, Russia, New Zealand, Lithuania, Malta,
 Slovakia, Israel, Croatia, Hungary, Ukraine, Estonia, Taiwan, Japan, Saudi Arabia, United States, South Korea,
 Czech Republic, United Kingdom, Kuwait, Finland, Canada, Australia, Slovenia, Latvia, Greece, Cyprus,
 Germany, Spain, Bahrain, France, Sweden, Denmark, Iceland, Netherlands, Portugal, Belgium, Italy, Norway,
 Austria, Macao, Hong Kong, Singapore, Switzerland, Ireland, Luxembourg, United Arab Emirates, Brunei, Qatar}
```

In[∘]:=
```
(ExploitTable = Table[{CountryList[[i]], ExploitationIndex[[1, i]], DataNoHeader[[i, 5]]}, {i, 1, N}]) // TableForm
Export["./ExploitTable2019.csv", ExploitTable, "CSV"]
```

Out[∘]//TableForm=

Malawi	1.13295	1305.19
Burundi	1.13023	1631.02
Sierra Leone	1.12934	2170.71
Congo - Kinshasa	1.1265	3118.09
Mali	1.12224	3686.45
Madagascar	1.12473	3775.58
Mozambique	1.11991	3830.02
Rwanda	1.1252	4038.81
Uganda	1.12737	4124.79
Burkina Faso	1.11894	4250.45
Zimbabwe	1.12706	4751.51
Liberia	1.1218	5063.44
Gambia	1.12026	5115.19
Niger	1.11247	5582.85
Ethiopia	1.11545	5788.34
Togo	1.11906	5899.5

Central African Republic	1.11382	6706.24
Tanzania	1.11477	7050.08
Sudan	1.11224	7417.71
Benin	1.11592	7457.46
Venezuela	1.12239	7641.56
Pakistan	1.11286	7934.12
Cameroon	1.10972	9657.7
Ivory Coast	1.10586	9923.93
Senegal	1.10442	9927.59
Kenya	1.11338	10 277.8
Nepal	1.10199	12 069.4
Cambodia	1.10259	12 763.3
Lesotho	1.09634	13 163.6
Kyrgyzstan	1.11613	13 657.1
Haiti	1.09498	13 846.3
Ghana	1.10775	13 847.8
Myanmar	1.09377	15 236.2
Nigeria	1.0947	15 942.3
Zambia	1.10345	17 004.9
Bangladesh	1.0936	17 438.7
Egypt	1.10117	18 151.1
Bolivia	1.10353	18 725.2
Yemen	1.08421	18 810.6
Nicaragua	1.09016	20 560.4
Vietnam	1.09878	20 829.6
Mauritania	1.07643	21 571.6
Syria	1.08947	23 881.5
Honduras	1.08558	24 197.5
Philippines	1.08825	25 466.5
Belize	1.09978	25 502.6
Guatemala	1.07019	25 643.4
India	1.07599	25 851.6
Congo – Brazzaville	1.07146	26 282.9
Laos	1.06728	26 688.5
Eswatini	1.07183	27 557.6
Moldova	1.09498	28 714.3

Iraq	1.07275	28 799.8
Tunisia	1.07969	30 061.1
El Salvador	1.0701	30 306.1
Namibia	1.06501	32 005.6
Fiji	1.07611	32 098.
Armenia	1.08197	33 453.7
Jordan	1.07412	35 686.4
Tajikistan	1.07251	38 304.6
Sri Lanka	1.06797	38 838.5
Paraguay	1.0594	40 950.9
Morocco	1.03096	41 995.8
Angola	0.999419	43 089.
Colombia	1.05311	43 595.5
Peru	1.05717	45 118.8
Costa Rica	1.05043	46 775.6
South Africa	1.052	49 484.1
Jamaica	1.03913	51 615.6
Mongolia	1.04893	54 044.3
Trinidad and Tobago	1.04675	56 047.7
Algeria	1.01988	57 776.4
Gabon	1.03602	59 112.6
Kazakhstan	1.03889	61 914.4
Brazil	1.03419	64 806.1
Bulgaria	1.03603	65 467.5
Ecuador	1.0199	66 934.3
Indonesia	0.998288	67 154.9
China	1.0113	70 822.5
Dominican Republic	1.01205	72 126.7
Argentina	1.01944	75 363.2
Mauritius	0.999177	76 687.6
Albania	1.00849	79 773.1
Thailand	1.00042	80 862.4
Botswana	1.00586	80 902.3
Iran	0.984658	82 308.1
Poland	1.02076	83 091.7
Mexico	0.992077	85 702.7

Serbia	1.01416	88 816.6
Uruguay	0.982677	92 135.7
Romania	1.00184	93 253.2
Chile	0.976184	109 529.
Maldives	0.944714	112 392.
Malaysia	0.969257	112 623.
Barbados	0.950517	118 077.
Turkey	0.9239	122 687.
Panama	0.94572	123 541.
Russia	0.960804	133 157.
New Zealand	0.955007	137 288.
Lithuania	0.949787	137 415.
Malta	0.9382	141 781.
Slovakia	0.967598	142 448.
Israel	0.967681	143 939.
Croatia	0.954984	145 845.
Hungary	0.944326	147 450.
Ukraine	0.923317	161 065.
Estonia	0.936674	165 543.
Taiwan	0.906679	180 486.
Japan	0.894906	205 761.
Saudi Arabia	0.835889	206 994.
United States	0.899167	209 865.
South Korea	0.892801	217 949.
Czech Republic	0.884313	222 218.
United Kingdom	0.885706	226 605.
Kuwait	0.76948	229 910.
Finland	0.866503	231 589.
Canada	0.877336	233 116.
Australia	0.866954	234 061.
Slovenia	0.865302	237 622.
Latvia	0.835101	241 613.
Greece	0.830534	244 976.
Cyprus	0.81157	246 638.
Germany	0.860228	250 343.
Spain	0.811982	253 067.

```
Bahrain                    0.732118   261 212.
France                     0.81773    267 067.
Sweden                     0.826464   273 383.
Denmark                    0.834457   276 318.
Iceland                    0.81348    276 431.
Netherlands                0.820439   277 550.
Portugal                   0.740576   284 953.
Belgium                    0.781844   302 980.
Italy                      0.775851   311 409.
Norway                     0.804695   320 338.
Austria                    0.78501    321 097.
Macao                      0.739533   337 297.
Hong Kong                  0.758752   345 093.
Singapore                  0.811127   370 239.
Switzerland                0.768917   375 453.
Ireland                    0.725552   382 714.
Luxembourg                 0.717684   448 055.
United Arab Emirates       0.634108   461 630.
Brunei                     0.63912    461 943.
Qatar                      0.620679   574 633.
```

Out[•]= ./ExploitTable2019.csv

In[•]:=
```
(* ExploitationArray=ArrayPlot[Transpose[ExploitationIndex],
  FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)
```

In[•]:=
```
(* Export["./ExploitationArray.eps",ExploitationArray,"EPS"] *)
```

In[•]:=
```
ExploitationIndexDistrib = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  ExploitationIndexDistrib[[t, i]] = {t, ExploitationIndex[[t, i]]}
 ]
]
```

Below the Gini coefficient for the exploitation intensity index is plotted over *T*. The pattern of the Gini coefficient is consistent with the previous charts depicting the exploitation intensity index.

In[∘]:=
```
(* Gini=Table[0.0,{T}];
For[t=1,t≤T,t++,
 Gini[[t]]=N/(N-1) ((Σ_{i=1}^N Σ_{j=1}^N Abs[ExploitationIndex[[t,i]]-ExploitationIndex[[t,j]]])/(2 N^2 Mean[ExploitationIndex[[t]]]))
] *)
```

In[∘]:=
```
Clear[SortExploitationIndex];
SortExploitationIndex = Table[Table[0.0, {N}], {t, 1, T}];
GiniTest = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortExploitationIndex[[t]] = Sort[ExploitationIndex[[t]]];
 GiniTest[[t]] = N/(N - 1) Σ_{i=1}^N ((2 i - N - 1) SortExploitationIndex[[t, i]]) / (N^2 Mean[SortExploitationIndex[[t]]])
]
```

In[∘]:=
```
Chop[GiniTest]
```

Out[∘]=
```
{0.0702262}
```

In[∘]:=
```
(* ExploitationGini=ListLinePlot[GiniTest,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: e_t^y"}] *)
```

In[∘]:=
```
(* Export["./ExploitationGini.eps",ExploitationGini,"EPS"] *)
```

Plotting Gini coefficient for wealth $W_{t-1} = p_{t-1}\,\omega_{t-1}$.

```
In[•]:= Clear[SortWealth, WealthGini];
SortWealth = Table[Table[0.0, {N}], {t, 1, T}];
WealthGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 SortWealth[[t]] = If[t == 1, Sort[AgentSet[[All, 1]]], Sort[Activities[[t - 1, All, 5]]]];
 WealthGini[[t]] = N/(N - 1) Sum[(2 i - N - 1) SortWealth[[t, i]]/(N^2 Mean[SortWealth[[t]]]), {i, 1, N}]
]
(* WealthGiniPlot=
 ListLinePlot[WealthGini,PlotStyle→{Thick,Blue},AxesLabel→{"t","Gini: W^γ_{t-1}"},LabelStyle→12,PlotRange→{0,1}] *)

In[•]:= (* Export["./WealthGini.eps",WealthGiniPlot,"EPS"] *)

In[•]:= WealthGini

Out[•]= {0.813248}
```

The figure below plots the distribution of wealth for select *t*.

```
In[•]:= (* WealthDistribRow=GraphicsRow[{
    Histogram[AgentSet[[All,1]],10,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 1",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[24,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 25",18],AxesLabel→{"ω^γ_{t-1}"}],
    Histogram[Activities[[49,All,5]],20,"Probability",AxesOrigin→{0,0},PlotRange→{0,1},
     ImageSize→{250,300},LabelStyle→12,PlotLabel→Style["t = 50",18],AxesLabel→{"ω^γ_{t-1}"}]
   },Spacings→{15,-20}
  ]
  Export["./WealthDistribRow.eps",WealthDistribRow,"EPS"]
*)
```

Income Figures

Figures on net and gross income.

In[•]:=
```
Clear[Income];
Income = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  Income⟦t, i⟧ = Profit⟦t⟧ × Activities⟦t, i, 5⟧ + Wage⟦t⟧ × Activities⟦t, i, 6⟧
 ]
]

IncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
```

$$\text{IncomeGini}[[t]] = \frac{N}{N-1}\left(\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\text{Abs}[\text{Income}[[t, i]] - \text{Income}[[t, j]]]\right) \Big/ \left(2\,N^2\,\text{Mean}[\text{Income}[[t]]]\right)\right)$$

```
]
(* IncomeGiniPlot=ListLinePlot[IncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[IncomeGini]+0.1},AxesLabel→{"t","Gini: Net Income"},LabelStyle→14] *)
```

In[•]:=
```
(* Export["./NetIncomeGini.eps",IncomeGiniPlot,"EPS"] *)
```

```
In[•]:= Clear[IncomeShares];
IncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  IncomeShares[[t, i]] = Income[[t, i]] / Total[Income[[t]]]
 ]
]
(* IncomeArray=ArrayPlot[Transpose[IncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},
  PlotLegends→Automatic,ColorFunction→"BlueGreenYellow",ColorFunctionScaling→True,
  DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./NetIncomeArray.eps",IncomeArray,"EPS"] *)

In[•]:= Clear[GrossIncome];
GrossIncome = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncome[[t, i]] = (1 + Profit[[t]]) Activities[[t, i, 5]] + Wage[[t]] × Activities[[t, i, 6]]
 ]
]

GrossIncomeGini = Table[0.0, {T}];
For[t = 1, t ≤ T, t++,
 GrossIncomeGini[[t]] = N/(N - 1) ((Sum[Sum[Abs[GrossIncome[[t, i]] - GrossIncome[[t, j]]], {j, 1, N}], {i, 1, N}]) / (2 N^2 Mean[GrossIncome[[t]]]))
]
(* GrossIncomeGiniPlot=ListLinePlot[GrossIncomeGini,PlotStyle→{Thick,Blue},
  PlotRange→{0,Max[GrossIncomeGini]+0.1},AxesLabel→{"t","Gini: Income"},LabelStyle→14] *)
```

```
In[•]:= (* Export["./GrossIncomeGini.eps",GrossIncomeGiniPlot,"EPS"] *)

In[•]:= GrossIncomeGini

Out[•]= {0.793429}

In[•]:= Clear[GrossIncomeShares];
GrossIncomeShares = Table[Table[0.0, {N}], {T}];
For[t = 1, t ≤ T, t++,
 For[i = 1, i ≤ N, i++,
  GrossIncomeShares[[t, i]] = GrossIncome[[t, i]] / Total[GrossIncome[[t]]]
 ]
]
(* GrossIncomeArray=
 ArrayPlot[Transpose[GrossIncomeShares],FrameLabel→{"ν(ω0 per capita)","t"},PlotLegends→Automatic,
  ColorFunctionScaling→True,DataReversed→True,FrameTicks→Automatic,AspectRatio→1.5,LabelStyle→18] *)

In[•]:= (* Export["./GrossIncomeArray.eps",GrossIncomeArray,"EPS"] *)
```

Updated Simulation Reporting

```
In[•]:= (ExploitedTable = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1.0,
      {CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploitedTable2019.csv", ExploitedTable, "CSV"]

Out[•]//TableForm=
Malawi                  &   1.13295   \\
Burundi                 &   1.13023   \\
Sierra Leone            &   1.12934   \\
Congo - Kinshasa        &   1.1265    \\
Mali                    &   1.12224   \\
Madagascar              &   1.12473   \\
Mozambique              &   1.11991   \\
```

```
Rwanda                    &   1.1252    \\
Uganda                    &   1.12737   \\
Burkina Faso              &   1.11894   \\
Zimbabwe                  &   1.12706   \\
Liberia                   &   1.1218    \\
Gambia                    &   1.12026   \\
Niger                     &   1.11247   \\
Ethiopia                  &   1.11545   \\
Togo                      &   1.11906   \\
Central African Republic  &   1.11382   \\
Tanzania                  &   1.11477   \\
Sudan                     &   1.11224   \\
Benin                     &   1.11592   \\
Venezuela                 &   1.12239   \\
Pakistan                  &   1.11286   \\
Cameroon                  &   1.10972   \\
Ivory Coast               &   1.10586   \\
Senegal                   &   1.10442   \\
Kenya                     &   1.11338   \\
Nepal                     &   1.10199   \\
Cambodia                  &   1.10259   \\
Lesotho                   &   1.09634   \\
Kyrgyzstan                &   1.11613   \\
Haiti                     &   1.09498   \\
Ghana                     &   1.10775   \\
Myanmar                   &   1.09377   \\
Nigeria                   &   1.0947    \\
Zambia                    &   1.10345   \\
Bangladesh                &   1.0936    \\
Egypt                     &   1.10117   \\
Bolivia                   &   1.10353   \\
Yemen                     &   1.08421   \\
Nicaragua                 &   1.09016   \\
Vietnam                   &   1.09878   \\
Mauritania                &   1.07643   \\
Syria                     &   1.08947   \\
```

```
Honduras                  &   1.08558   \\
Philippines               &   1.08825   \\
Belize                    &   1.09978   \\
Guatemala                 &   1.07019   \\
India                     &   1.07599   \\
Congo - Brazzaville       &   1.07146   \\
Laos                      &   1.06728   \\
Eswatini                  &   1.07183   \\
Moldova                   &   1.09498   \\
Iraq                      &   1.07275   \\
Tunisia                   &   1.07969   \\
El Salvador               &   1.0701    \\
Namibia                   &   1.06501   \\
Fiji                      &   1.07611   \\
Armenia                   &   1.08197   \\
Jordan                    &   1.07412   \\
Tajikistan                &   1.07251   \\
Sri Lanka                 &   1.06797   \\
Paraguay                  &   1.0594    \\
Morocco                   &   1.03096   \\
Colombia                  &   1.05311   \\
Peru                      &   1.05717   \\
Costa Rica                &   1.05043   \\
South Africa              &   1.052     \\
Jamaica                   &   1.03913   \\
Mongolia                  &   1.04893   \\
Trinidad and Tobago       &   1.04675   \\
Algeria                   &   1.01988   \\
Gabon                     &   1.03602   \\
Kazakhstan                &   1.03889   \\
Brazil                    &   1.03419   \\
Bulgaria                  &   1.03603   \\
Ecuador                   &   1.0199    \\
China                     &   1.0113    \\
Dominican Republic        &   1.01205   \\
Argentina                 &   1.01944   \\
```

```
Albania                    &    1.00849    \\
Thailand                   &    1.00042    \\
Botswana                   &    1.00586    \\
Poland                     &    1.02076    \\
Serbia                     &    1.01416    \\
Romania                    &    1.00184    \\
```

Out[•]= ./ExploitedTable2019.csv

In[•]:=
```
(ExploiterTable = DeleteCases[Table[If[ExploitationIndex⟦1, i⟧ < 1.0,
      {CountryList⟦i⟧, "&", ExploitationIndex⟦1, i⟧, "\\\\"},], {i, 1, N}], Null]) // TableForm
Export["./ExploiterTable2019.csv", ExploiterTable, "CSV"]
```

Out[•]//TableForm=
```
Angola                 &    0.999419    \\
Indonesia              &    0.998288    \\
Mauritius              &    0.999177    \\
Iran                   &    0.984658    \\
Mexico                 &    0.992077    \\
Uruguay                &    0.982677    \\
Chile                  &    0.976184    \\
Maldives               &    0.944714    \\
Malaysia               &    0.969257    \\
Barbados               &    0.950517    \\
Turkey                 &    0.9239      \\
Panama                 &    0.94572     \\
Russia                 &    0.960804    \\
New Zealand            &    0.955007    \\
Lithuania              &    0.949787    \\
Malta                  &    0.9382      \\
Slovakia               &    0.967598    \\
Israel                 &    0.967681    \\
Croatia                &    0.954984    \\
Hungary                &    0.944326    \\
Ukraine                &    0.923317    \\
Estonia                &    0.936674    \\
Taiwan                 &    0.906679    \\
Japan                  &    0.894906    \\
```

```
Saudi Arabia           &   0.835889   \\
United States          &   0.899167   \\
South Korea            &   0.892801   \\
Czech Republic         &   0.884313   \\
United Kingdom         &   0.885706   \\
Kuwait                 &   0.76948    \\
Finland                &   0.866503   \\
Canada                 &   0.877336   \\
Australia              &   0.866954   \\
Slovenia               &   0.865302   \\
Latvia                 &   0.835101   \\
Greece                 &   0.830534   \\
Cyprus                 &   0.81157    \\
Germany                &   0.860228   \\
Spain                  &   0.811982   \\
Bahrain                &   0.732118   \\
France                 &   0.81773    \\
Sweden                 &   0.826464   \\
Denmark                &   0.834457   \\
Iceland                &   0.81348    \\
Netherlands            &   0.820439   \\
Portugal               &   0.740576   \\
Belgium                &   0.781844   \\
Italy                  &   0.775851   \\
Norway                 &   0.804695   \\
Austria                &   0.78501    \\
Macao                  &   0.739533   \\
Hong Kong              &   0.758752   \\
Singapore              &   0.811127   \\
Switzerland            &   0.768917   \\
Ireland                &   0.725552   \\
Luxembourg             &   0.717684   \\
United Arab Emirates   &   0.634108   \\
Brunei                 &   0.63912    \\
Qatar                  &   0.620679   \\
```

Out[•]= ./ExploiterTable2019.csv

In[•]:=
```
(ExploitIncomeCSV = Table[{CountryList[[i]], "&", ExploitationIndex[[1, i]], "\\\\"}, {i, 1, N}]) // TableForm
Export["./ExploitIncomeTable2019.csv", ExploitIncomeCSV, "CSV"]
```

Out[•]//TableForm=

Malawi	&	1.13295	\\
Burundi	&	1.13023	\\
Sierra Leone	&	1.12934	\\
Congo - Kinshasa	&	1.1265	\\
Mali	&	1.12224	\\
Madagascar	&	1.12473	\\
Mozambique	&	1.11991	\\
Rwanda	&	1.1252	\\
Uganda	&	1.12737	\\
Burkina Faso	&	1.11894	\\
Zimbabwe	&	1.12706	\\
Liberia	&	1.1218	\\
Gambia	&	1.12026	\\
Niger	&	1.11247	\\
Ethiopia	&	1.11545	\\
Togo	&	1.11906	\\
Central African Republic	&	1.11382	\\
Tanzania	&	1.11477	\\
Sudan	&	1.11224	\\
Benin	&	1.11592	\\
Venezuela	&	1.12239	\\
Pakistan	&	1.11286	\\
Cameroon	&	1.10972	\\
Ivory Coast	&	1.10586	\\
Senegal	&	1.10442	\\
Kenya	&	1.11338	\\
Nepal	&	1.10199	\\
Cambodia	&	1.10259	\\
Lesotho	&	1.09634	\\
Kyrgyzstan	&	1.11613	\\
Haiti	&	1.09498	\\

```
Ghana                    &   1.10775    \\
Myanmar                  &   1.09377    \\
Nigeria                  &   1.0947     \\
Zambia                   &   1.10345    \\
Bangladesh               &   1.0936     \\
Egypt                    &   1.10117    \\
Bolivia                  &   1.10353    \\
Yemen                    &   1.08421    \\
Nicaragua                &   1.09016    \\
Vietnam                  &   1.09878    \\
Mauritania               &   1.07643    \\
Syria                    &   1.08947    \\
Honduras                 &   1.08558    \\
Philippines              &   1.08825    \\
Belize                   &   1.09978    \\
Guatemala                &   1.07019    \\
India                    &   1.07599    \\
Congo - Brazzaville      &   1.07146    \\
Laos                     &   1.06728    \\
Eswatini                 &   1.07183    \\
Moldova                  &   1.09498    \\
Iraq                     &   1.07275    \\
Tunisia                  &   1.07969    \\
El Salvador              &   1.0701     \\
Namibia                  &   1.06501    \\
Fiji                     &   1.07611    \\
Armenia                  &   1.08197    \\
Jordan                   &   1.07412    \\
Tajikistan               &   1.07251    \\
Sri Lanka                &   1.06797    \\
Paraguay                 &   1.0594     \\
Morocco                  &   1.03096    \\
Angola                   &   0.999419   \\
Colombia                 &   1.05311    \\
Peru                     &   1.05717    \\
Costa Rica               &   1.05043    \\
```

```
South Africa              &   1.052      \\
Jamaica                   &   1.03913    \\
Mongolia                  &   1.04893    \\
Trinidad and Tobago       &   1.04675    \\
Algeria                   &   1.01988    \\
Gabon                     &   1.03602    \\
Kazakhstan                &   1.03889    \\
Brazil                    &   1.03419    \\
Bulgaria                  &   1.03603    \\
Ecuador                   &   1.0199     \\
Indonesia                 &   0.998288   \\
China                     &   1.0113     \\
Dominican Republic        &   1.01205    \\
Argentina                 &   1.01944    \\
Mauritius                 &   0.999177   \\
Albania                   &   1.00849    \\
Thailand                  &   1.00042    \\
Botswana                  &   1.00586    \\
Iran                      &   0.984658   \\
Poland                    &   1.02076    \\
Mexico                    &   0.992077   \\
Serbia                    &   1.01416    \\
Uruguay                   &   0.982677   \\
Romania                   &   1.00184    \\
Chile                     &   0.976184   \\
Maldives                  &   0.944714   \\
Malaysia                  &   0.969257   \\
Barbados                  &   0.950517   \\
Turkey                    &   0.9239     \\
Panama                    &   0.94572    \\
Russia                    &   0.960804   \\
New Zealand               &   0.955007   \\
Lithuania                 &   0.949787   \\
Malta                     &   0.9382     \\
Slovakia                  &   0.967598   \\
Israel                    &   0.967681   \\
```

```
Croatia                 &   0.954984   \\
Hungary                 &   0.944326   \\
Ukraine                 &   0.923317   \\
Estonia                 &   0.936674   \\
Taiwan                  &   0.906679   \\
Japan                   &   0.894906   \\
Saudi Arabia            &   0.835889   \\
United States           &   0.899167   \\
South Korea             &   0.892801   \\
Czech Republic          &   0.884313   \\
United Kingdom          &   0.885706   \\
Kuwait                  &   0.76948    \\
Finland                 &   0.866503   \\
Canada                  &   0.877336   \\
Australia               &   0.866954   \\
Slovenia                &   0.865302   \\
Latvia                  &   0.835101   \\
Greece                  &   0.830534   \\
Cyprus                  &   0.81157    \\
Germany                 &   0.860228   \\
Spain                   &   0.811982   \\
Bahrain                 &   0.732118   \\
France                  &   0.81773    \\
Sweden                  &   0.826464   \\
Denmark                 &   0.834457   \\
Iceland                 &   0.81348    \\
Netherlands             &   0.820439   \\
Portugal                &   0.740576   \\
Belgium                 &   0.781844   \\
Italy                   &   0.775851   \\
Norway                  &   0.804695   \\
Austria                 &   0.78501    \\
Macao                   &   0.739533   \\
Hong Kong               &   0.758752   \\
Singapore               &   0.811127   \\
Switzerland             &   0.768917   \\
```

```
Ireland                    &    0.725552    \\
Luxembourg                 &    0.717684    \\
United Arab Emirates       &    0.634108    \\
Brunei                     &    0.63912     \\
Qatar                      &    0.620679    \\
```

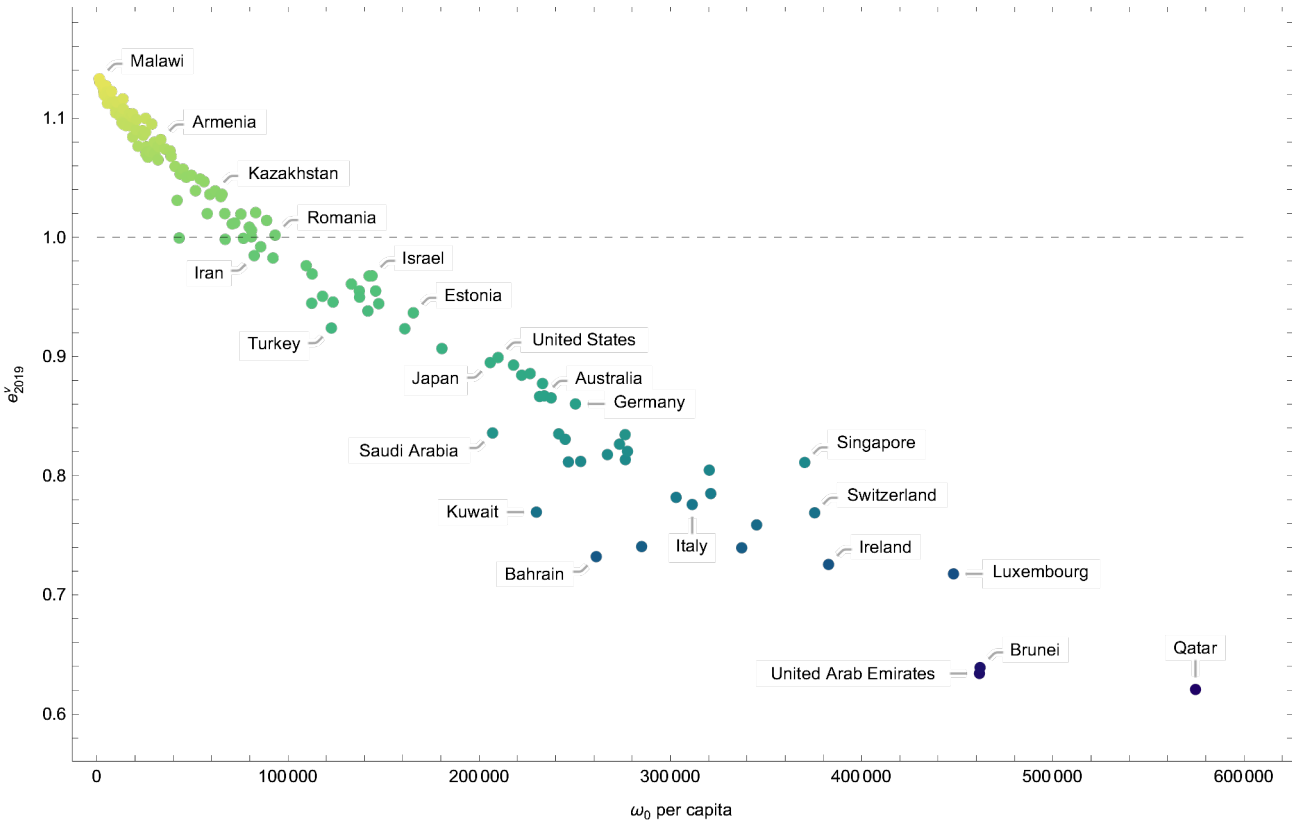
Out[•]= ./ExploitIncomeTable2019.csv

In[•]:= **DataNoHeader[[1]]**

Out[•]= {Malawi, 24 314., 2.0391, 18.6287, 1305.19}

In[∘]:=
```
ExploitWealthPlot = ListPlot[Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, 1, N}] → CountryList,
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_2019"}, ColorFunction → "BlueGreenYellow",
  Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitWealthPlot2019.eps", ExploitWealthPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploitWealthPlot2019.eps

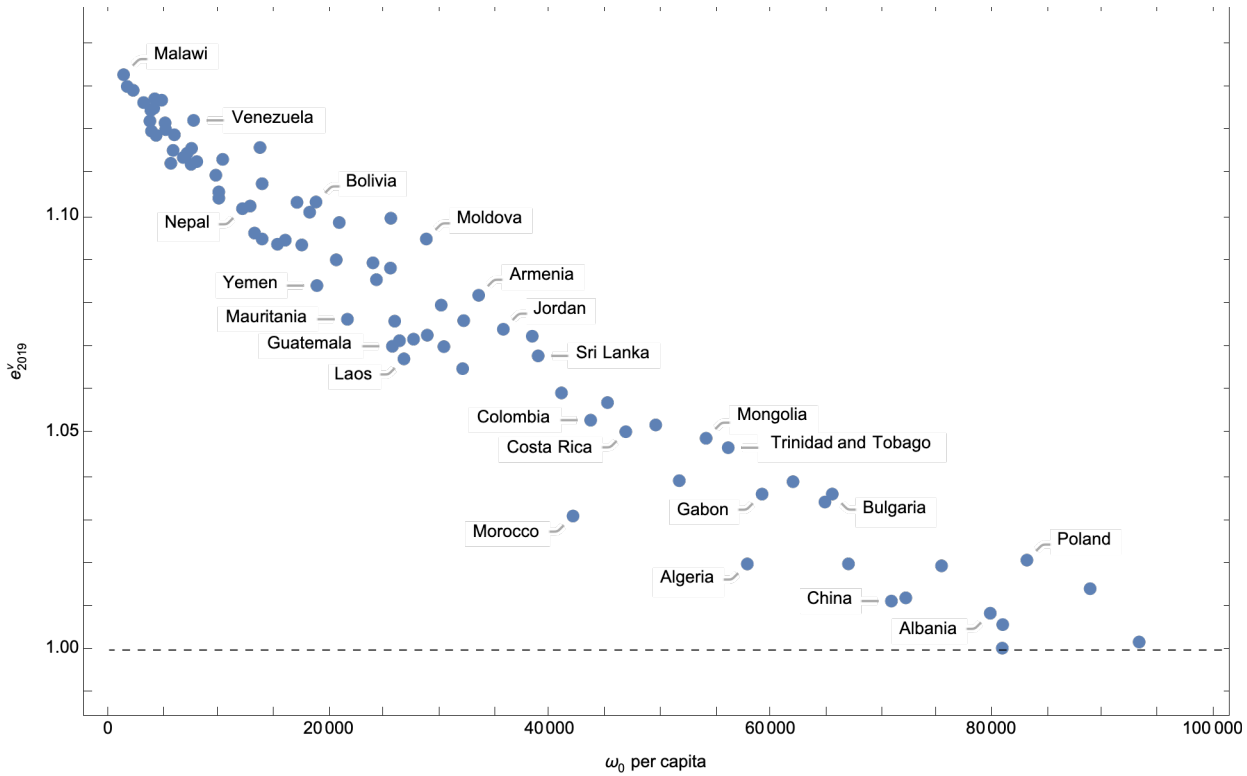
In[∘]:=
```
ExploitedCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] > 1, i, 0.], {i, 1, N}], 0.]
ExploiterCountries = DeleteCases[Table[If[ExploitationIndex[[1, i]] < 1, i, 0.], {i, 1, N}], 0.]
```

Out[∘]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30,
 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
 59, 60, 61, 62, 63, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 79, 80, 81, 83, 84, 85, 87, 89, 91}
```

Out[∘]=
```
{64, 78, 82, 86, 88, 90, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106,
 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125,
 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144}
```

In[∘]:=
```
ExploitedCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploitedCountries}] → CountryList[[ExploitedCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_2019"}, Epilog → {Black, Dashed, Line[{{0, 1}, {600 000, 1}}]}]
Export["./ExploitedCountriesPlot2019.eps", ExploitedCountriesPlot, "EPS"]
```

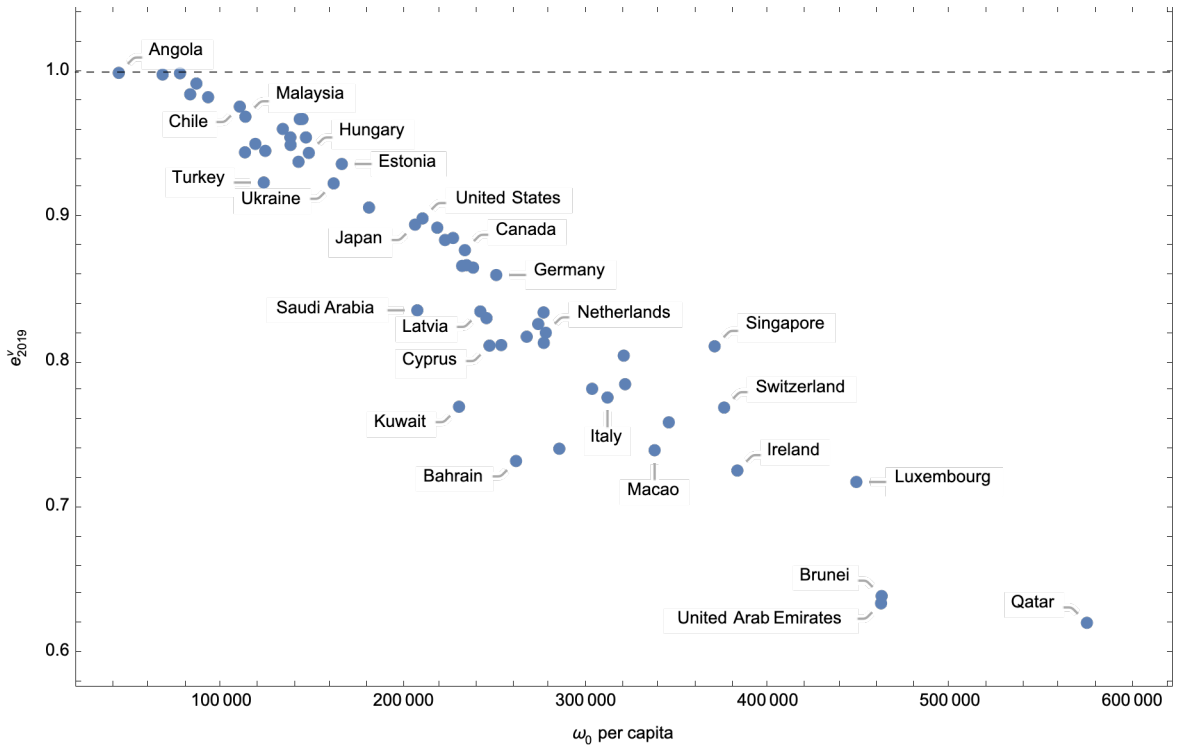
Out[∘]=



Out[∘]= ./ExploitedCountriesPlot2019.eps

In[∘]:=
```
ExploiterCountriesPlot = ListPlot[
  Table[{DataNoHeader[[i, 5]], ExploitationIndex[[1, i]]}, {i, ExploiterCountries}] → CountryList[[ExploiterCountries]],
  LabelingFunction → Callout[Automatic, Automatic], PlotRange → All, Frame → True,
  FrameLabel → {"ω0 per capita", "e^v_2019"}, Epilog → {Black, Dashed, Line[{{0, 1}, {800 000, 1}}]}]
Export["./ExploiterCountriesPlot2019.eps", ExploiterCountriesPlot, "EPS"]
```

Out[∘]=



Out[∘]= ./ExploiterCountriesPlot2019.eps

In[∘]:=
```
(outlierTesting = Table[{CountryList[[i]], AgentSet[[i, 1]], AgentSet[[i, 2]]}, {i, 1, Length[CountryList]}]) // TableForm
Export["./outlierTesting2019.csv", outlierTesting, "CSV"];
```

Out[∘]//TableForm=

Malawi	24 314.	3.79858×10^6
Burundi	18 806.6	1.63331×10^6

Sierra Leone	16 960.2	1.28535×10^6
Congo – Kinshasa	270 621.	1.4573×10^7
Mali	72 468.2	2.71359×10^6
Madagascar	101 825.	4.63926×10^6
Mozambique	116 302.	3.73016×10^6
Rwanda	50 998.1	2.42312×10^6
Uganda	182 603.	1.07885×10^7
Burkina Faso	86 375.1	2.61374×10^6
Zimbabwe	69 588.2	3.97391×10^6
Liberia	25 000.2	907 642.
Gambia	12 008.9	393 662.
Niger	130 140.	2.85556×10^6
Ethiopia	648 750	1.63097×10^7
Togo	47 682.1	1.4533×10^6
Central African Republic	31 822.4	741 010.
Tanzania	397 483.	9.67254×10^6
Sudan	317 576.	6.90063×10^6
Benin	88 006.9	2.26418×10^6
Venezuela	217 905.	8.25105×10^6
Pakistan	1 718 255	3.83494×10^7
Cameroon	249 906.	4.90565×10^6
Ivory Coast	255 209.	4.36023×10^6
Senegal	161 784.	2.63594×10^6
Kenya	540 345.	1.23375×10^7
Nepal	345 289.	5.21708×10^6
Cambodia	210 422.	3.23745×10^6
Lesotho	27 976.7	361 195.
Kyrgyzstan	87 622.6	2.27752×10^6
Haiti	155 952	1.94469×10^6
Ghana	421 222.	7.6863×10^6
Myanmar	823 446.	9.96489×10^6
Nigeria	3.20382×10^6	3.96742×10^7

Zambia	303 724.	4.79889×10^{6}
Bangladesh	2.84331×10^{6}	3.42691×10^{7}
Egypt	1.82215×10^{6}	2.68719×10^{7}
Bolivia	215 586.	3.41479×10^{6}
Yemen	548 553.	5.37454×10^{6}
Nicaragua	134 578.	1.4965×10^{6}
Vietnam	2.00926×10^{6}	2.76846×10^{7}
Mauritania	97 626.7	826 302.
Syria	407 660.	4.46332×10^{6}
Honduras	235 831.	2.37561×10^{6}
Philippines	2 753 356	2.93526×10^{7}
Belize	9956.21	141 044.
Guatemala	450 849.	3.43613×10^{6}
India	35 324 124	2.96677×10^{8}
Congo – Brazzaville	141 415.	1.10031×10^{6}
Laos	191 343.	1.39303×10^{6}
Eswatini	31 638.9	247 645.
Moldova	116 100.	1.44774×10^{6}
Iraq	1.13211×10^{6}	8.99762×10^{6}
Tunisia	351 555.	3.15663×10^{6}
El Salvador	195 583.	1.48865×10^{6}
Namibia	79 837.9	561 612.
Fiji	28 567.3	240 442.
Armenia	98 945.9	927 535.
Jordan	360 494.	2.93172×10^{6}
Tajikistan	357 037.	2.82631×10^{6}
Sri Lanka	828 180.	6.09431×10^{6}
Paraguay	288 483.	1.87147×10^{6}
Morocco	1.53166×10^{6}	7.06021×10^{6}
Angola	1.37132×10^{6}	4.71651×10^{6}
Colombia	2.19457×10^{6}	1.30822×10^{7}
Peru	1.46683×10^{6}	9.22811×10^{6}

Costa Rica	236 105.	1.36023×10^{6}
South Africa	2 897 706	1.70299×10^{7}
Jamaica	152 178.	766 411.
Mongolia	174 304.	985 460.
Trinidad and Tobago	78 186.6	430 399.
Algeria	2.48746×10^{6}	1.02639×10^{7}
Gabon	128 428.	624 905.
Kazakhstan	1.1486×10^{6}	5.76949×10^{6}
Brazil	13 677 290	6.52438×10^{7}
Bulgaria	458 279.	2.23023×10^{6}
Ecuador	1.1629×10^{6}	4.79948×10^{6}
Indonesia	18 173 848	6.19245×10^{7}
China	101 544 168	3.86978×10^{8}
Dominican Republic	774 569.	2.9717×10^{6}
Argentina	3.37482×10^{6}	1.38677×10^{7}
Mauritius	97 370.3	334 223.
Albania	229 818.	854 187.
Thailand	5 630 091	1.95258×10^{7}
Botswana	186 375.	677 011.
Iran	6 824 485	2.08752×10^{7}
Poland	3 148 161	1.31008×10^{7}
Mexico	10 933 568	3.5429×10^{7}
Serbia	621 228.	2.42975×10^{6}
Uruguay	318 946.	961 106.
Romania	1.80581×10^{6}	6.33803×10^{6}
Chile	2 075 795	5.96192×10^{6}
Maldives	59 680.2	138 644.
Malaysia	3.5983×10^{6}	9.83607×10^{6}
Barbados	33 888.2	81 683.1
Turkey	10 235 724	2.09767×10^{7}
Panama	524 603.	1.22645×10^{6}
Russia	19 423 886	5.00984×10^{7}
New Zealand	656 663.	1.62965×10^{6}

Lithuania	379 210.	909 757.
Malta	62 440.5	139 321.
Slovakia	777 340.	2.1004×10^6
Israel	1.22627×10^6	3.31532×10^6
Croatia	602 383.	1.49471×10^6
Hungary	1.42801×10^6	3.30936×10^6
Ukraine	7.08584×10^6	1.44726×10^7
Estonia	219 444.	485 103.
Taiwan	4 258 740	7.92283×10^6
Japan	26 102 876	4.55936×10^7
Saudi Arabia	7.09338×10^6	9.29979×10^6
United States	69 059 088	1.23376×10^8
South Korea	11 164 507	1.92868×10^7
Czech Republic	2.37533×10^6	3.92753×10^6
United Kingdom	15 302 674	2.54832×10^7
Kuwait	967 252.	956 063.
Finland	1.2812×10^6	1.93876×10^6
Canada	8 721 109	1.3921×10^7
Australia	5 899 094	8.94638×10^6
Slovenia	493 945.	743 114.
Latvia	460 683.	601 831.
Greece	2 565 751	3.28376×10^6
Cyprus	214 205.	252 247.
Germany	20 907 856	3.06958×10^7
Spain	11 827 519	1.39528×10^7
Bahrain	428 702.	365 906.
France	17 987 252	2.17551×10^7
Sweden	2 743 785	3.44851×10^6
Denmark	1 594 880	2.07748×10^6
Iceland	93 710.	111 267.
Netherlands	4.7453×10^6	5.80771×10^6
Portugal	2.91399×10^6	2.56964×10^6

Belgium	3 496 175	3.63373×10^6
Italy	18 855 818	1.91241×10^7
Norway	1.72307×10^6	1.97018×10^6
Austria	2.87546×10^6	3.02772×10^6
Macao	216 005.	189 712.
Hong Kong	2 566 180	2.43008×10^6
Singapore	2 148 978	2.5258×10^6
Switzerland	3 225 664	3.18114×10^6
Ireland	1.8686×10^6	1.55532×10^6
Luxembourg	275 867.	222 859.
United Arab Emirates	4.51036×10^6	2.68366×10^6
Brunei	200 160.	121 246.
Qatar	1.62742×10^6	923 123.

Checking across years

In[∘]:=
```
e1970 = Import["./ExploitedTable1970.csv", "CSV"];
e1980 = Import["./ExploitedTable1980.csv", "CSV"];
e1990 = Import["./ExploitedTable1990.csv", "CSV"];
e2000 = Import["./ExploitedTable2000.csv", "CSV"];
e2010 = Import["./ExploitedTable2010.csv", "CSV"];
e2019 = Import["./ExploitedTable2019.csv", "CSV"];
```

In[∘]:=
```
ex1970 = Import["./ExploiterTable1970.csv", "CSV"];
ex1980 = Import["./ExploiterTable1980.csv", "CSV"];
ex1990 = Import["./ExploiterTable1990.csv", "CSV"];
ex2000 = Import["./ExploiterTable2000.csv", "CSV"];
ex2010 = Import["./ExploiterTable2010.csv", "CSV"];
ex2019 = Import["./ExploiterTable2019.csv", "CSV"];
```

In[∘]:=
```
SortBy[ex1970, 1] // TableForm
```

Out[∘]//TableForm=

Algeria	&	0.851501	\\
Angola	&	0.92645	\\

```
Australia              &   0.848898    \\
Austria                &   0.971583    \\
Bahrain                &   0.86663     \\
Belgium                &   0.830314    \\
Brunei                 &   0.36121     \\
Canada                 &   0.884936    \\
Colombia               &   0.967709    \\
Cyprus                 &   0.833119    \\
Denmark                &   0.858385    \\
Finland                &   0.842174    \\
France                 &   0.816647    \\
Germany                &   0.857475    \\
Ghana                  &   0.987495    \\
Greece                 &   0.887339    \\
Iceland                &   0.767       \\
Iran                   &   0.934046    \\
Israel                 &   0.924842    \\
Italy                  &   0.874912    \\
Japan                  &   0.975076    \\
Kuwait                 &   0.871663    \\
Luxembourg             &   0.670182    \\
Mexico                 &   0.978653    \\
Netherlands            &   0.815456    \\
New Zealand            &   0.97004     \\
Norway                 &   0.85793     \\
Portugal               &   0.952563    \\
Qatar                  &   0.746596    \\
Saudi Arabia           &   0.84659     \\
Spain                  &   0.947657    \\
Sweden                 &   0.851425    \\
Switzerland            &   0.752133    \\
Turkey                 &   0.977488    \\
United Arab Emirates   &   0.0992267   \\
United Kingdom         &   0.898705    \\
United States          &   0.820437    \\
Venezuela              &   0.937686    \\
```

In[∘]:=
```
Position[{ex1970, ex1980}, "USSR"]
```

Out[∘]=
```
{}
```

In[∘]:=
```
{SortBy[e1970, 1], SortBy[e1980, 1]} // TableForm
```

Out[∘]//TableForm=

Albania	Argentina	Bangladesh	Barbados	Belize	Benin	Bolivia	Botswana	Brazil	Bulgaria	Bu
&	&	&	&	&	&	&	&	&	&	&
1.04913	1.09181	1.12435	1.05917	1.11186	1.1118	1.09551	1.11219	1.07406	1.11968	1
\\	\\	\\	\\	\\	\\	\\	\\	\\	\\	\
Albania	Argentina	Bangladesh	Barbados	Belize	Benin	Bolivia	Botswana	Brazil	Bulgaria	Bu
&	&	&	&	&	&	&	&	&	&	&
1.05397	1.08883	1.12457	1.02482	1.10629	1.11003	1.10323	1.09047	1.04121	1.1103	1
\\	\\	\\	\\	\\	\\	\\	\\	\\	\\	\

In[∘]:=
```
Position[{ex1990, ex2000, ex2010, ex2019}, "Russia"]
```

Out[∘]=
```
{{1, 29, 1}, {2, 17, 1}, {3, 19, 1}, {4, 13, 1}}
```

In[∘]:=
```
DeleteDuplicates[Join[e1970[[All, 1]], e1980[[All, 1]]]]
```

Out[∘]=
```
{Rwanda, Myanmar, Mali, Egypt, Sierra Leone, Nepal, Ethiopia, Burkina Faso, Mozambique, Malawi,
 El Salvador, Vietnam, Maldives, Eswatini, Bangladesh, Uganda, China, Lesotho, Burundi, Ivory Coast,
 Laos, Madagascar, Morocco, Benin, Sudan, Botswana, Pakistan, Zimbabwe, Indonesia, Tanzania, Bulgaria,
 Cameroon, Gambia, Congo - Brazzaville, Togo, Congo - Kinshasa, India, Paraguay, Kenya, Sri Lanka,
 Jordan, Honduras, Central African Republic, Tunisia, Mongolia, Haiti, Iraq, Belize, Liberia, Guatemala,
 Peru, Cambodia, Zambia, Romania, Thailand, Mauritius, Bolivia, Fiji, Senegal, Philippines, Argentina,
 Brazil, Dominican Republic, Panama, Syria, Niger, Taiwan, South Korea, Namibia, Costa Rica, Albania,
 Malta, Gabon, Macao, Nicaragua, Mauritania, Nigeria, Trinidad and Tobago, Malaysia, Poland, Ecuador,
 Chile, Barbados, Singapore, Uruguay, Hungary, South Africa, Hong Kong, Jamaica, Ireland, Ghana, Colombia}
```

Join exploiter and exploited sets for each year to get full country lists for generating tables.

In[∘]:=
```
comb1970 = Join[e1970, ex1970];
comb1980 = Join[e1980, ex1980];
comb1990 = Join[e1990, ex1990];
comb2000 = Join[e2000, ex2000];
comb2010 = Join[e2010, ex2010];
comb2019 = Join[e2019, ex2019];
```

In[∘]:=
```
comb1970[[All, 1]]
```

Out[∘]=
```
{Rwanda, Myanmar, Mali, Egypt, Sierra Leone, Nepal, Ethiopia, Burkina Faso, Mozambique, Malawi, El Salvador,
 Vietnam, Maldives, Eswatini, Bangladesh, Uganda, China, Lesotho, Burundi, Ivory Coast, Laos, Madagascar,
 Morocco, Benin, Sudan, Botswana, Pakistan, Zimbabwe, Indonesia, Tanzania, Bulgaria, Cameroon, Gambia,
 Congo - Brazzaville, Togo, Congo - Kinshasa, India, Paraguay, Kenya, Sri Lanka, Jordan, Honduras,
 Central African Republic, Tunisia, Mongolia, Haiti, Iraq, Belize, Liberia, Guatemala, Peru, Cambodia, Zambia,
 Romania, Thailand, Mauritius, Bolivia, Fiji, Senegal, Philippines, Argentina, Brazil, Dominican Republic,
 Panama, Syria, Niger, Taiwan, South Korea, Namibia, Costa Rica, Albania, Malta, Gabon, Macao, Nicaragua,
 Mauritania, Nigeria, Trinidad and Tobago, Malaysia, Poland, Ecuador, Chile, Barbados, Singapore, Uruguay,
 Hungary, South Africa, Hong Kong, Jamaica, Ireland, Ghana, Turkey, Angola, Iran, Portugal, Mexico, Venezuela,
 Colombia, Algeria, Bahrain, Spain, Austria, Kuwait, Japan, Saudi Arabia, New Zealand, Greece, Israel,
 Italy, Cyprus, United Kingdom, Canada, Qatar, Finland, Belgium, Denmark, Norway, France, Sweden, Germany,
 Australia, Netherlands, Iceland, United States, Luxembourg, Switzerland, Brunei, United Arab Emirates}
```

In[∘]:=
```
Complement[comb1980[[All, 1]], comb1970[[All, 1]]]
```

Out[∘]=
```
{}
```

In[∘]:=
```
postSoviet = Complement[comb1990[[All, 1]], comb1980[[All, 1]]]
```

Out[∘]=
```
{Armenia, Croatia, Czech Republic, Estonia, Kazakhstan, Kyrgyzstan, Latvia,
 Lithuania, Moldova, Russia, Serbia, Slovakia, Slovenia, Tajikistan, Ukraine, Yemen}
```

In[∘]:=
```
Complement[comb2000[[All, 1]], comb1990[[All, 1]]]
```

Out[∘]=
```
{}
```

In[∘]:=
```
Complement[comb2010[[All, 1]], comb2000[[All, 1]]]
```
Out[∘]= { }

In[∘]:=
```
Complement[comb2019[[All, 1]], comb2010[[All, 1]]]
```
Out[∘]= { }

Generating tables

In[∘]:=
```
sort1970 = SortBy[comb1970, 1];
sort1980 = SortBy[comb1980, 1];
sort1990 = SortBy[comb1990, 1];
sort2000 = SortBy[comb2000, 1];
sort2010 = SortBy[comb2010, 1];
sort2019 = SortBy[comb2019, 1];
```

In[∘]:=
```
Length[sort1970]
Length[sort1980]
Length[sort1990]
Length[sort2000]
Length[sort2010]
Length[sort2019]
```
Out[∘]= 128

Out[∘]= 128

Out[∘]= 144

Out[∘]= 144

Out[∘]= 144

Out[∘]= 144

Generate combined tables pre- and post-1989.

```
In[•]:= combTab1 = Table[Join[sort1970⟦i, 1 ;; 3⟧, sort1980⟦i, 2 ;; 4⟧], {i, 1, Length[sort1970]}];
        Export["./combTab1.csv", combTab1, "CSV"];

In[•]:= combTab2 = Table[
           Join[sort1990⟦i, 1 ;; 3⟧, sort2000⟦i, 2 ;; 3⟧, sort2010⟦i, 2 ;; 3⟧, sort2019⟦i, 2 ;; 4⟧], {i, 1, Length[sort1990]}];
        Export["./combTab2.csv", combTab2, "CSV"];

In[•]:= Export["./postSovietList.csv", postSoviet, "CSV"];

In[•]:= Length[postSoviet]

Out[•]= 16

In[•]:= Flatten[Table[Position[combTab2⟦All, 1⟧, postSoviet⟦i⟧], {i, 1, Length[postSoviet]}]]

Out[•]= {5, 31, 33, 39, 67, 70, 72, 75, 87, 109, 113, 116, 117, 127, 135, 142}
```

Labour Values and Productivity

Estimating Labour Values

```
In[•]:= Clear[t, T];
        years = Table[1997 + i, {i, 0, 23}]

Out[•]= {1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007,
         2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020}

In[•]:= Length[years]

Out[•]= 24
```

Read in labour requirement matrix.

```
In[•]:= lData = Drop[Import["./IO Data/l1test2.xlsx", "XLSX"]⟦1⟧, 2];
```

Read in price deflators.

```
In[•]:= priceDeflatorsRAW = Drop[Import["./IO Data/inputDeflators.xls", "XLS"][[1]], 7];

In[•]:= priceDeflatorsRAW2 = priceDeflatorsRAW[[3 ;; 16]];

In[•]:= priceDeflators = Table[Table[priceDeflatorsRAW2[[j, t]], {j, 1, 14}], {t, 3, 2 + Length[years]}];

In[•]:= priceDeflators[[1 ;; 3]]

Out[•]= {{53.299, 50.406, 54.432, 66.697, 62.619, 77.123, 76.582, 60.525, 88.102, 75.665, 76.042, 70.742, 71.469, 78.747},
  {50.814, 45.601, 50.674, 66.51, 60.063, 76.932, 77.214, 59.095, 88.34, 75.986, 76.938, 71.456, 72.221, 78.754},
  {49.89, 49.754, 52.808, 67.816, 60.019, 77.697, 78.42, 61.037, 88.854, 76.541, 78.307, 73.171, 73.435, 79.066}}

In[•]:= priceDeflatorsMatrix = Table[Table[Table[0.0, {14}], {14}], {Length[years]}];
```

$$\text{In[•]:= priceDeflatorsMatrix = Table}\left[\text{Table}\left[\text{Table}\left[\frac{\frac{\text{priceDeflators}[[t,i]]}{100.}}{\frac{\text{priceDeflators}[[t,j]]}{100.}}, \{i, 1, 14\}\right], \{j, 1, 14\}\right], \{t, 1, \text{Length[years]}\}\right];$$

In[ ]:=
```
priceDeflatorsMatrix[[1]]
```

Out[ ]=
```
{{1., 0.945721, 1.02126, 1.25137, 1.17486, 1.44699, 1.43684, 1.13557,
  1.65298, 1.41963, 1.42671, 1.32727, 1.34091, 1.47746}, {1.05739, 1., 1.07987, 1.3232,
  1.24229, 1.53004, 1.5193, 1.20075, 1.74785, 1.50111, 1.50859, 1.40344, 1.41787, 1.56225},
 {0.979185, 0.926036, 1., 1.22533, 1.15041, 1.41687, 1.40693, 1.11194, 1.61857, 1.39008, 1.39701,
  1.29964, 1.313, 1.4467}, {0.799121, 0.755746, 0.816109, 1., 0.938858, 1.15632, 1.14821,
  0.907462, 1.32093, 1.13446, 1.14011, 1.06065, 1.07155, 1.18067}, {0.851163, 0.804963, 0.869257,
  1.06512, 1., 1.23162, 1.22298, 0.96656, 1.40695, 1.20834, 1.21436, 1.12972, 1.14133, 1.25756},
 {0.691091, 0.653579, 0.705782, 0.864813, 0.811937, 1., 0.992985, 0.784785, 1.14236, 0.981095,
  0.985983, 0.917262, 0.926689, 1.02106}, {0.695973, 0.658196, 0.710768, 0.870923, 0.817673,
  1.00706, 1., 0.790329, 1.15043, 0.988026, 0.992949, 0.923742, 0.933235, 1.02827},
 {0.880611, 0.832813, 0.899331, 1.10197, 1.0346, 1.27423, 1.2653, 1., 1.45563, 1.25014,
  1.25637, 1.16881, 1.18082, 1.30107}, {0.604969, 0.572132, 0.617829, 0.757043, 0.710756,
  0.875383, 0.869242, 0.686988, 1., 0.858834, 0.863113, 0.802956, 0.811207, 0.893816},
 {0.704408, 0.666173, 0.719381, 0.881478, 0.827582, 1.01927, 1.01212, 0.799907, 1.16437, 1., 1.00498,
  0.934937, 0.944545, 1.04073}, {0.700915, 0.662871, 0.715815, 0.877107, 0.823479, 1.01422, 1.0071,
  0.795942, 1.1586, 0.995042, 1., 0.930302, 0.939862, 1.03557}, {0.753428, 0.712533, 0.769444,
  0.94282, 0.885174, 1.0902, 1.08255, 0.855574, 1.2454, 1.06959, 1.07492, 1., 1.01028, 1.11316},
 {0.745764, 0.705285, 0.761617, 0.93323, 0.87617, 1.07911, 1.07154, 0.846871, 1.23273, 1.05871,
  1.06399, 0.989828, 1., 1.10183}, {0.676838, 0.640101, 0.691226, 0.846978, 0.795192,
  0.979377, 0.972507, 0.768601, 1.1188, 0.960862, 0.965649, 0.898345, 0.907577, 1.}}
```

Read in input-output tables.

In[ ]:=
```
ADataRaw = Table[0.0, {years}];
Table[ADataRaw[[t]] = Drop[Import["./IO Data/" <> ToString[years[[t]]] <> ".xls", "XLS"][[1]], 7], {t, 1, Length[years]}];
```

Define industries.

In[∘]:=
```
(industries = ADataRaw⟦1, 1 ;; 14, 2⟧) // TableForm
```

Out[∘]//TableForm=

```
Agriculture, forestry, fishing, and hunting
Mining
Utilities
Construction
Manufacturing
Wholesale trade
Retail trade
Transportation and warehousing
Information
Finance, insurance, real estate, rental, and leasing
Professional and business services
Educational services, health care, and social assistance
Arts, entertainment, recreation, accommodation, and food services
Other services, except government
```

Total industry output and value added in each year.

In[∘]:=
```
output = Table[ADataRaw⟦t, 23, 3 ;; 16⟧, {t, 1, Length[years]}];
```

In[∘]:=
```
valueAdded = Table[ADataRaw⟦t, 22, 3 ;; 16⟧, {t, 1, Length[years]}];
```

Read in value added deflators.

In[∘]:=
```
vaDeflatorsRAW = Drop[Import["./IO Data/vaDeflators.xls", "XLS"]⟦1⟧, 7];
```

In[∘]:=
```
vaDeflatorsRAW2 = vaDeflatorsRAW⟦3 ;; 16⟧;
```

In[∘]:=
```
vaDeflators = Table[Table[vaDeflatorsRAW2⟦j, t⟧, {j, 1, Length[industries]}], {t, 3, 2 + Length[years]}];
```

Selecting relevant input-output data on uses from raw IO matrix.

In[∘]:=
```
ASelect = Table[0.0, {years}];
Table[
  (ASelect⟦t⟧ = ADataRaw⟦t, 1 ;; 14, 3 ;; 16⟧ /. "---" → 0.), {t, 1, Length[years]}];
```

Scaling inputs used by total industry output to calculate a_{ij} and construct traditional A matrix.

In[•]:=
```
AMatrix = Table[0.0, {years}];
Table[
  AMatrix[[t]] = Table[ASelect[[t, All, j]] / output[[t, j]] priceDeflatorsMatrix[[t, All, j]], {j, 1, Length[industries]}],
  {t, 1, Length[years]}];
```

Scaling direct labour requirements by total industry output: $l_j = L_j / x_j$

In[•]:=
```
lData[[All, 2]]
```

Out[•]=
```
{1651., 654., 620.9, 5813., 17419., 5663.9, 14388.9, 4026.5, 3084., 7178., 14335., 14185., 11018., 4825.}
```

In[•]:=
```
lMatrix = Table[0.0, {years}];
lVectors = Table[0.0, {years}];
```

In[•]:=
```
Table[lMatrix[[t]] = lData[[All, t + 1]], {t, 1, Length[years]}];
Table[lVectors[[t]] = lMatrix[[t]] / output[[t]] / (priceDeflators[[t]] / 100.), {t, 1, Length[years]}];
```

Calculate labour values: v_j.

In[•]:=
```
lVectors[[1]].Inverse[IdentityMatrix[Length[industries]] - AMatrix[[1]]]
```

Out[•]=
```
{0.0202171, 0.0153283, 0.0102562, 0.0137925, 0.0702721, 0.0129341,
 0.0249233, 0.0191997, 0.00969162, 0.0267525, 0.0402894, 0.0222866, 0.0333928, 0.0222251}
```

In[•]:=
```
values = Table[0.0, {years}];
```

In[•]:=
```
Table[values[[t]] = lVectors[[t]].Inverse[IdentityMatrix[Length[industries]] - AMatrix[[t]]], {t, 1, Length[years]}];
```

Calculating labor productivity: $1 / v_j$

In[•]:=
```
1 / values[[1]]
```

Out[•]=
```
{49.4632, 65.2387, 97.502, 72.5034, 14.2304, 77.3153,
 40.1231, 52.0841, 103.182, 37.3796, 24.8204, 44.87, 29.9466, 44.9941}
```

In[∘]:=
```
laborProd = Table[0.0, {years}];
```

In[∘]:=
$$\text{Table}\left[\text{laborProd}[[t]] = \frac{1}{\text{values}[[t]]}, \{t, 1, \text{Length}[\text{years}]\}\right];$$

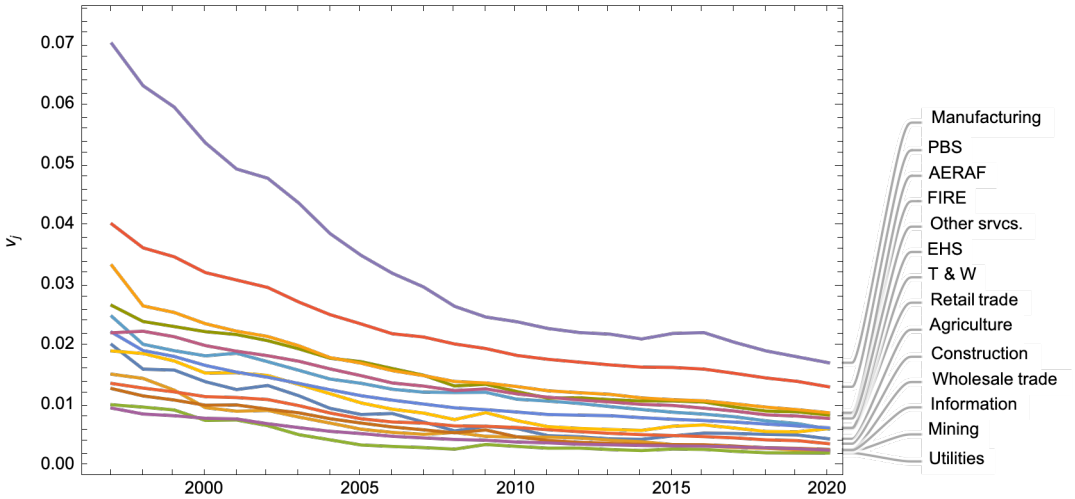
Labour Values Diagrams & Tables

In[∘]:=
```
industryLabels = {"Agriculture", "Mining", "Utilities", "Construction", "Manufacturing",
   "Wholesale trade", "Retail trade", "T & W", "Information", "FIRE", "PBS", "EHS", "AERAF", "Other srvcs."};
```

In[∘]:=
```
laborContent = Table[Table[{years[[t]], values[[t, j]]}, {j, 1, Length[industries]}], {t, 1, Length[years]}];
```

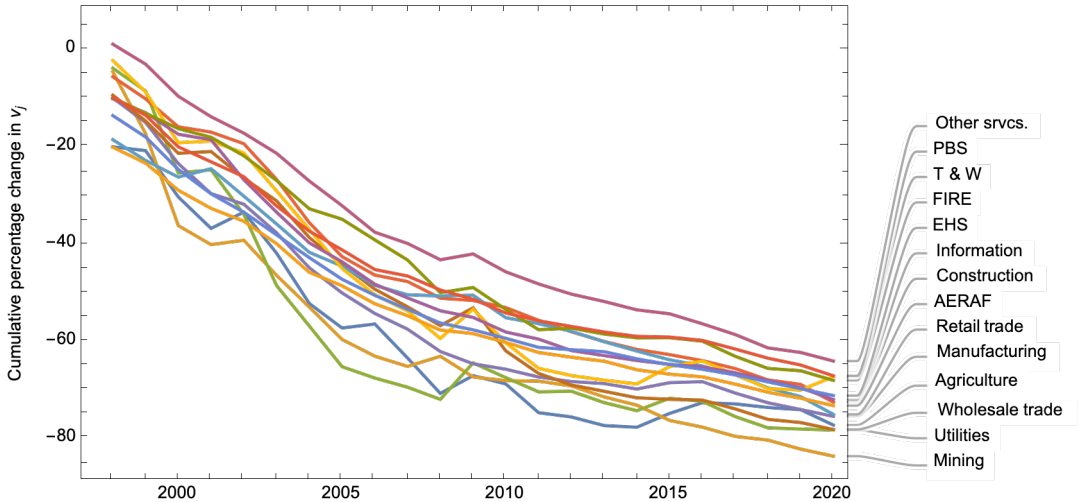
In[∘]:=
```
decreasingLabor = ListLinePlot[Table[laborContent[[All, j]], {j, 1, Length[industries]}],
  PlotLabels → industryLabels, ImageSize → Large, Frame → True, FrameLabel → {None, "vj"}, PlotRange → All]
Export["./LaborContentPlot.eps", decreasingLabor, "EPS"];
```

Out[∘]=



In[ ]:=
```
laborContent2 =
  Table[Table[{years[[t]], (values[[t, j]] - values[[1, j]])/values[[1, j]] 100}, {j, 1, Length[industries]}], {t, 2, Length[years]}];
decreasingLabor2 = ListLinePlot[Table[laborContent2[[All, j]], {j, 1, Length[industries]}], PlotLabels → industryLabels,
  ImageSize → Large, Frame → True, FrameLabel → {None, "Cumulative percentage change in v_j"}]
Export["./LaborContentChangePlot.eps", decreasingLabor2, "EPS"];
```

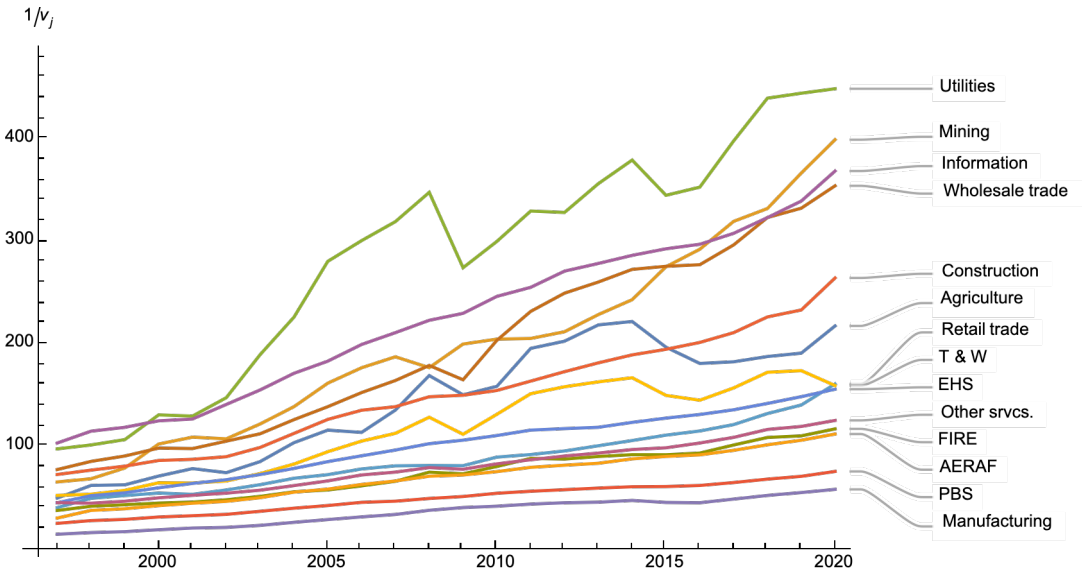
Out[ ]=



In[ ]:=
```
vProdYear = Table[Table[{years[[t]], laborProd[[t, j]]}, {j, 1, Length[industries]}], {t, 1, Length[years]}];
```

In[∘]:=
```
valueProdPlot = ListLinePlot[Table[vProdYear[[All, j]], {j, 1, Length[industries]}],
  PlotLabels → industryLabels, ImageSize → Large, AxesLabel → {None, "1/vj"}]
Export["./valueProdPlot.eps", valueProdPlot, "EPS"];
```

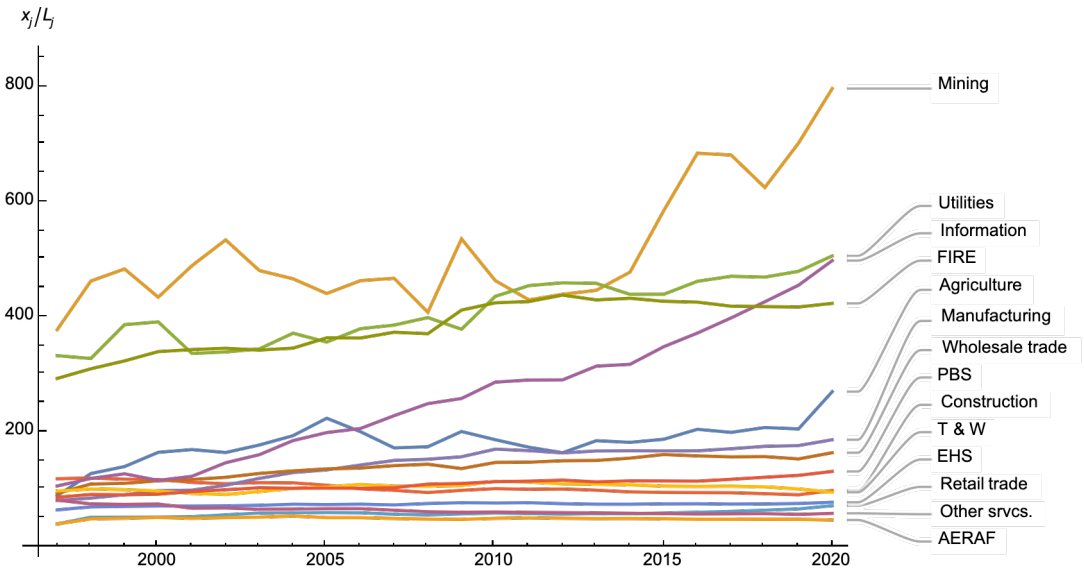
Out[∘]=



In[∘]:=
```
prodYear = Table[Table[{years[[t]], valueAdded[[t,j]]/((vaDeflators[[t,j]]/100.)) / lMatrix[[t, j]]}, {j, 1, Length[industries]}], {t, 1, Length[years]}];
```

In[∘]:=
```
priceProdPlot = ListLinePlot[Table[prodYear[[All, j]], {j, 1, Length[industries]}],
  PlotLabels → industryLabels, ImageSize → Large, AxesLabel → {None, "xj/Lj"}]
Export["./priceProdPlot.eps", priceProdPlot, "EPS"];
```

Out[∘]=



In[∘]:=
```
prodYear[[1]]
```

Out[∘]=
```
{{1997, 88.8621}, {1997, 377.453}, {1997, 332.461}, {1997, 118.494}, {1997, 80.7147}, {1997, 92.9862}, {1997, 39.9019},
 {1997, 97.0506}, {1997, 105.888}, {1997, 292.851}, {1997, 86.2376}, {1997, 64.3479}, {1997, 39.8953}, {1997, 81.3181}}
```

In[∘]:=
```
vProdYear[[1]]
```

Out[∘]=
```
{{1997, 49.4632}, {1997, 65.2387}, {1997, 97.502}, {1997, 72.5034}, {1997, 14.2304}, {1997, 77.3153}, {1997, 40.1231},
 {1997, 52.0841}, {1997, 103.182}, {1997, 37.3796}, {1997, 24.8204}, {1997, 44.87}, {1997, 29.9466}, {1997, 44.9941}}
```

In[∘]:=
```
prodComp = Table[Table[{prodYear[[t, j, 2]], vProdYear[[t, j, 2]]}, {t, 1, Length[years]}], {j, 1, Length[industries]}];
```

In[∘]:=
```
prodComp[[1]]
```

Out[∘]=
```
{{88.8621, 49.4632}, {127.509, 61.8197}, {139.666, 62.4236}, {164.385, 70.9701},
 {169.17, 78.2364}, {164.065, 74.293}, {177.151, 85.0958}, {193.359, 103.543}, {223.695, 115.783},
 {200.258, 113.589}, {172.158, 135.372}, {174.058, 169.068}, {200.611, 150.268}, {186.357, 158.538},
 {173.235, 195.591}, {163.727, 202.611}, {184.644, 218.5}, {181.715, 221.834}, {187.116, 196.741},
 {204.457, 180.875}, {199.045, 182.516}, {207.562, 187.647}, {205.111, 190.933}, {269.901, 217.529}}
```

In[∘]:=
```
ToString[industryLabels]
```

Out[∘]=
```
{Agriculture, Mining, Utilities, Construction, Manufacturing,
 Wholesale trade, Retail trade, T & W, Information, FIRE, PBS, EHS, AERAF, Other srvcs.}
```

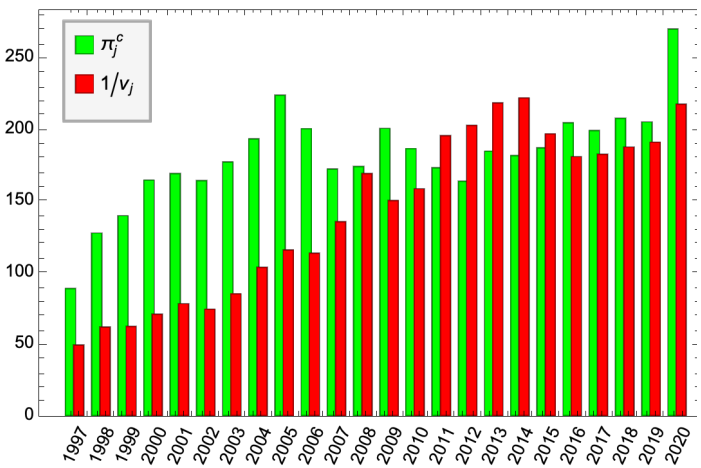
In[∘]:=
```
ToString[industryLabels[[1]]] <> "ProdPlot"
```

Out[∘]=
```
AgricultureProdPlot
```

In[∘]:=
```
prodCharts = Table[0.0, {Length[industries]}];
```

In[∘]:=
```
BarChart[prodComp[[1]], ChartLabels → {Placed[years, Axis, Rotate[#, 65 Degree] &], None},
 BarSpacing → {-0.2, 0.75}, ChartStyle → {Green, Red}, Frame → True, ImageSize → Medium,
 LegendAppearance → {LegendFunction → Panel}, ChartLegends → Placed[{Style["π_j^c", 10], Style["1/v_j", 10]}, {0.1, 0.85}]]
```

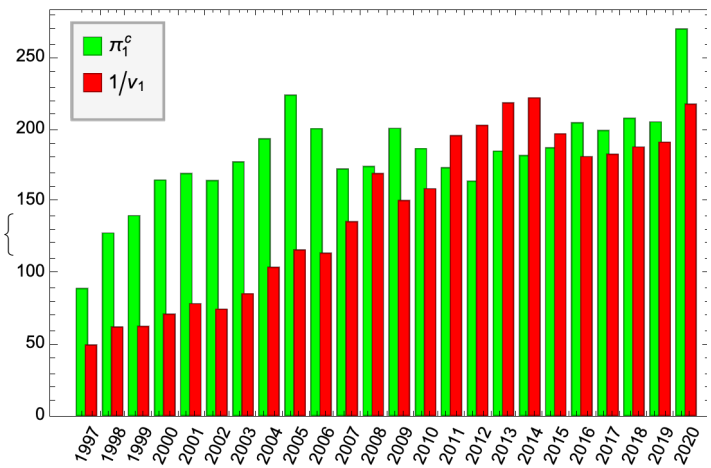
Out[∘]=



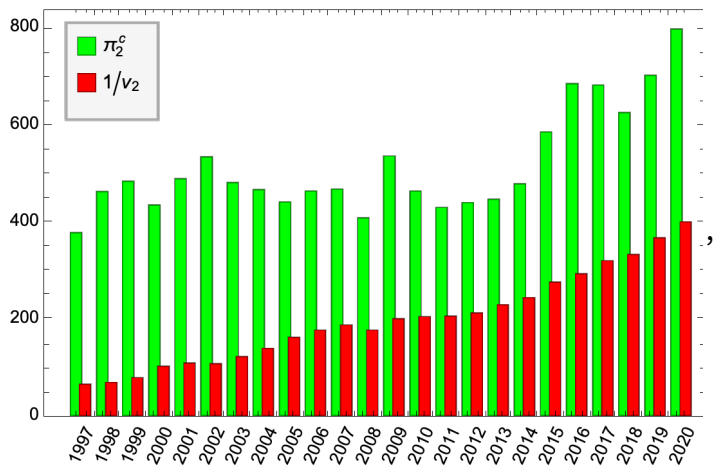
In[∘]:=
```
prodLegend = {{Style["π_1^c", 10], Style["1/v_1", 10]}, {Style["π_2^c", 10], Style["1/v_2", 10]},
   {Style["π_3^c", 10], Style["1/v_3", 10]}, {Style["π_4^c", 10], Style["1/v_4", 10]}, {Style["π_5^c", 10], Style["1/v_5", 10]},
   {Style["π_6^c", 10], Style["1/v_6", 10]}, {Style["π_7^c", 10], Style["1/v_7", 10]}, {Style["π_8^c", 10], Style["1/v_8", 10]},
   {Style["π_9^c", 10], Style["1/v_9", 10]}, {Style["π_10^c", 10], Style["1/v_10", 10]},
   {Style["π_11^c", 10], Style["1/v_11", 10]}, {Style["π_12^c", 10], Style["1/v_12", 10]},
   {Style["π_13^c", 10], Style["1/v_13", 10]}, {Style["π_14^c", 10], Style["1/v_14", 10]}};
```

In[∘]:=
```
Table[prodCharts[[j]] =
  BarChart[prodComp[[j]], ChartLabels → {Placed[years, Axis, Rotate[#, 65 Degree] &], None}, BarSpacing → {-0.2, 0.5},
   ChartStyle → {Green, Red}, Frame → True, ImageSize → Medium, LegendAppearance → {LegendFunction → Panel},
   ChartLegends → Placed[prodLegend[[j]], {0.1, 0.85}]], {j, 1, Length[industries]}]
Table[
  Export[ToString[industryLabels[[j]]] <> "ProdPlot.eps", prodCharts[[j]], "EPS"],
  {j, 1, Length[industries]}];
```

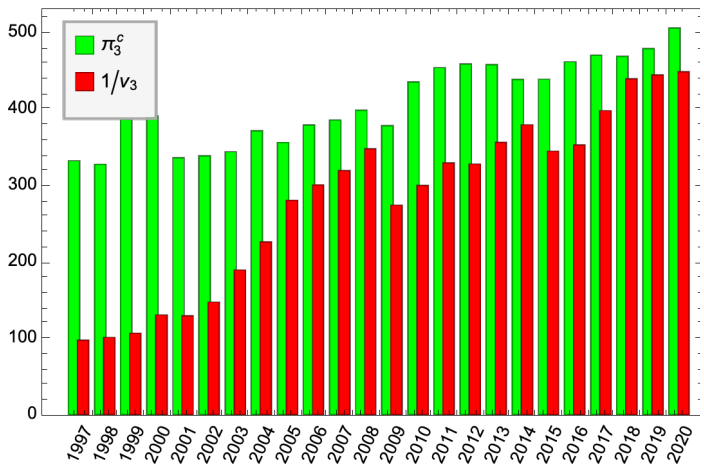
Out[∘]=



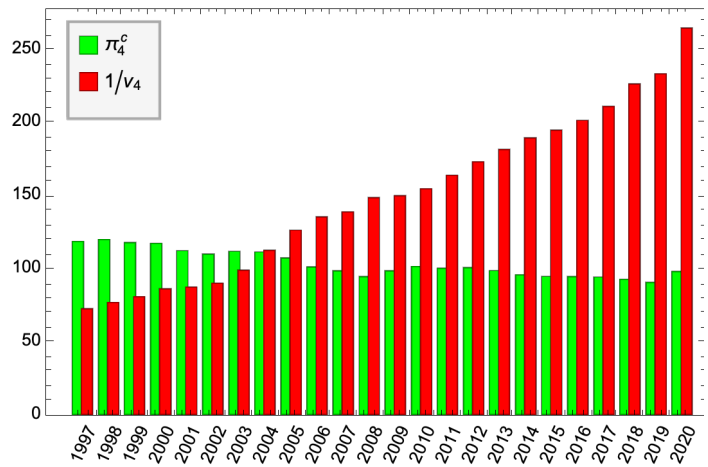
,



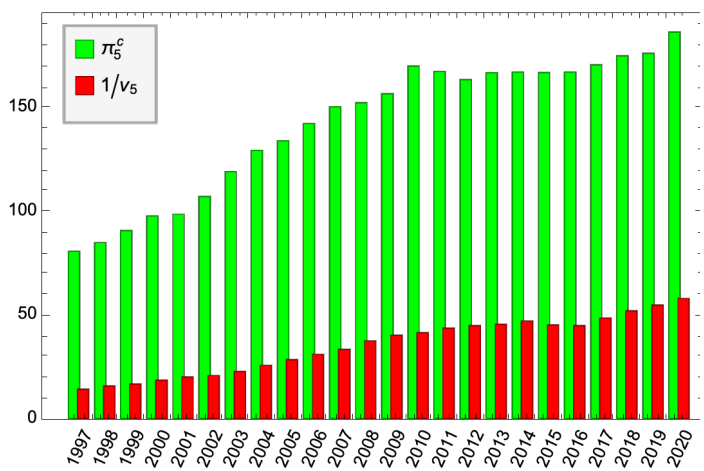
,


π_3^c
$1/v_3$
0
100
200
300
400
500
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

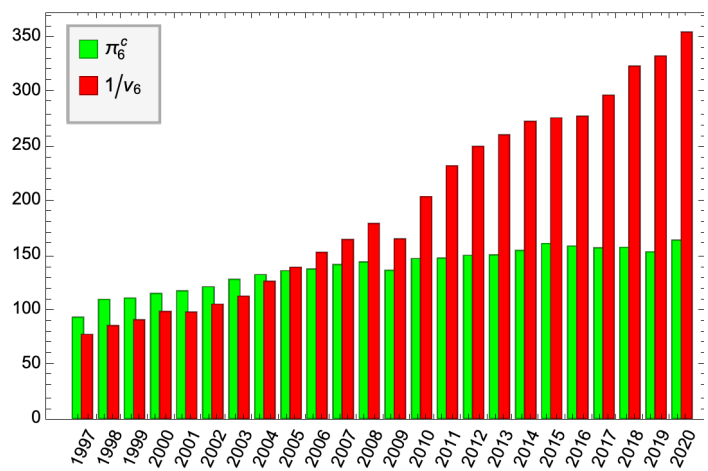
,


π_4^c
$1/v_4$
0
50
100
150
200
250
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

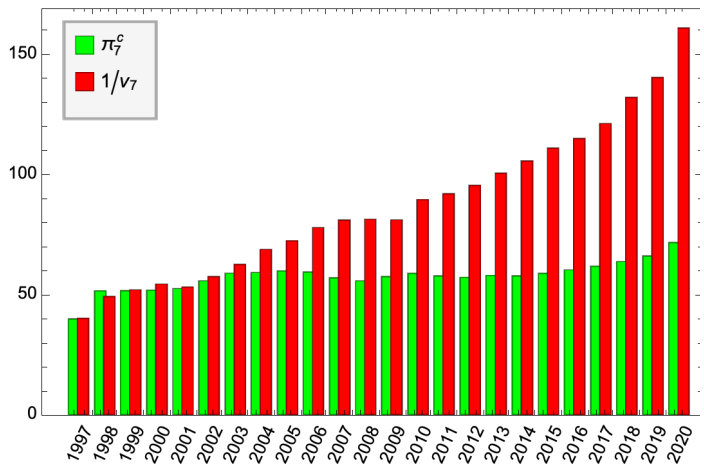
,


π_5^c
$1/v_5$
0
50
100
150
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

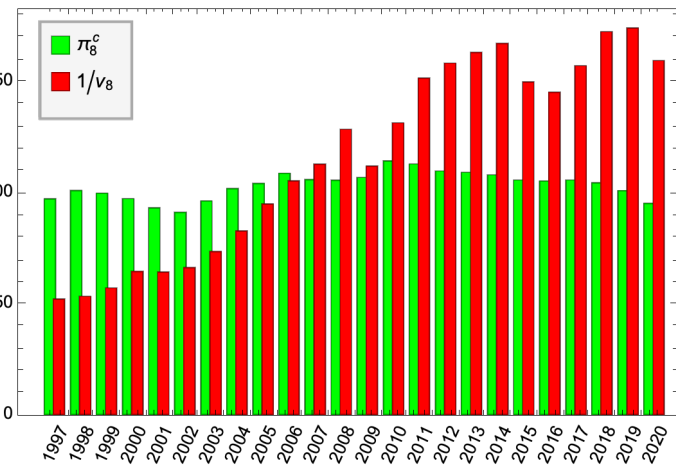
,


π_6^c
$1/v_6$
0
50
100
150
200
250
300
350
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

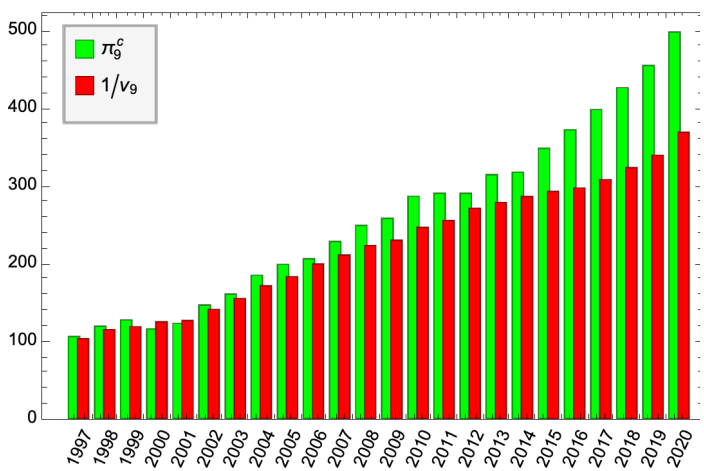
,


π_7^c
$1/v_7$
0
50
100
150
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

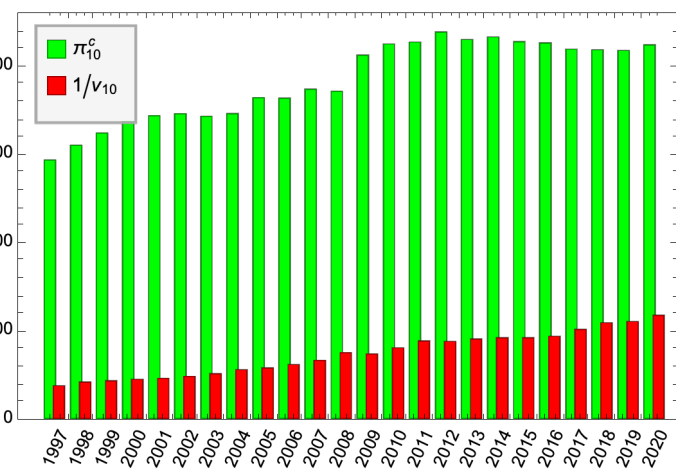
,


π_8^c
$1/v_8$
0
50
100
150
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

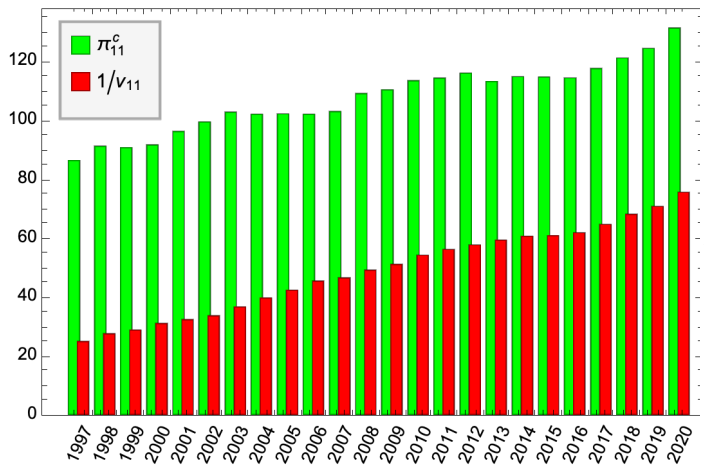
,


π_9^c
$1/v_9$
0
100
200
300
400
500
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

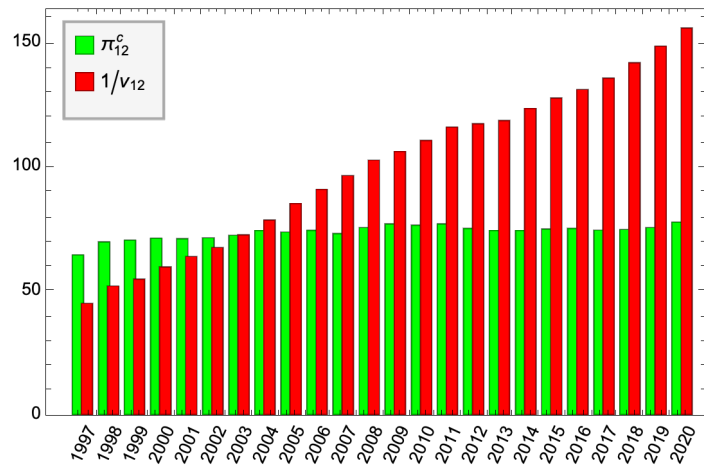
,


π_{10}^c
$1/v_{10}$
0
100
200
300
400
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020

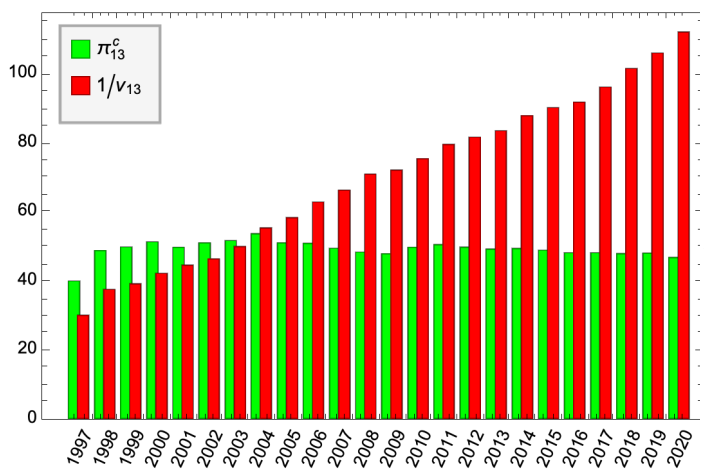
,


π_{11}^c
$1/v_{11}$

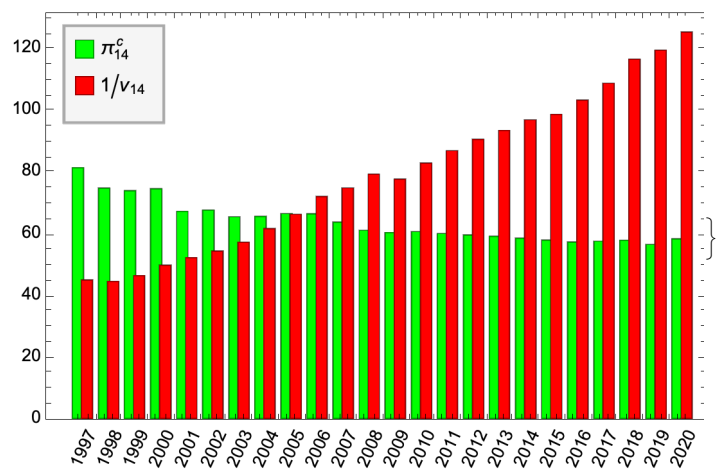
,


π_{12}^c
$1/v_{12}$

,


π_{13}^c
$1/v_{13}$

,


π_{14}^c
$1/v_{14}$

}