General Equilibrium Computational Exercise

In doing applied microeconomics you often have to compute equilibria of models that don't have closed-form solutions. The computation therefore must be done by iterative numerical methods. That's what you'll do in this exercise, for the Cobb-Douglas example in the semester's first lecture. The iterative computation is pretty straightforward, because there are only two goods and the demand functions have simple closed-form solutions. Moreover, the equilibrium itself has a closed-form solution, so you can also have your program compute the equilibrium prices directly and then check whether your iterative program converges to the correct equilibrium prices.

Specifically, you are to use a spreadsheet program such as Excel, or a programming language such as C+ or Pascal, to compute the path taken by prices and excess demands in the two-person, two-good, pure exchange Cobb-Douglas example from the first lecture, assuming that prices adjust according to the transition function in the Gale-Nikaido proof of existence of equilibrium:

$$f(p) = \frac{1}{\sum_{k=1}^{l} [p_k + M_k(p)]}[p + M(p)]$$

where $M_k(p) = \max\{0, \lambda z_k(p)\}$ for each good *k*.

Note that the proof did not actually include a λ, *i.e.*, we could assume that $\lambda = 1$. However, with $\lambda = 1$ the iterative process defined by this transition function will not converge for the Cobb-Douglas example, as you can verify once you've created your computational program. You'll find that to achieve convergence you'll need to use a λ equal to about .02 or smaller. Recall, too, that the proof does not actually apply to the Cobb-Douglas example, because demands are not defined for the price-lists (1,0) and (0,1). For the same reason, you can't start the iterative process off using either of these as the initial price-lists, because the "next *p*" defined by $f(p)$ won't be well-defined.

You will of course have to use specific parameter values for the two consumers' utility functions and endowment bundles. With a small enough value for λ, the process will converge for just about any parameter values and any strictly positive initial prices. Of course, when you run your program you should note whether it does converge to the equilibrium price-list.

Choose some specific parameter values and initial price-list, and plot (by hand) the price-line and the chosen bundles in the Edgeworth Box for several iterations of the process. Note that if the prices aren't sufficiently close to the equilibrium prices, the chosen bundles may not lie within the confines of the box. This is an important point to understand: each individual consumer simply takes the prices as given and chooses his or her best bundle within the resulting budget set. The consumer takes no account of the total resources available, nor of the other consumers' preferences or choices, because the consumer isn't assumed to have that information.

Cobb-Douglas utilities: $u(x,y) = x^a y^{1-a} = x^a y^b$, $a+b=1$

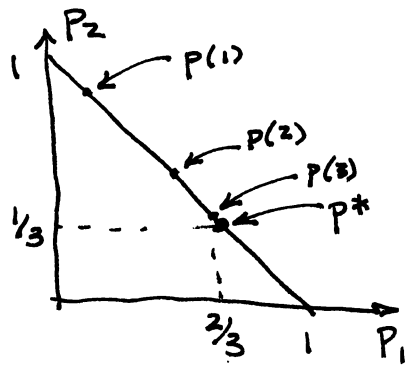
The example from the first lecture →

a1 =	0.875	a2 =	0.5	(a^i, x^i, y^i)
x1 =	40	x2 =	80	
y1 =	80	y2 =	40	
b1 =	0.125	b2 =	0.5	
a1*y1 =	70	a2*y2 =	20	
b1*x1 =	5	b2*x2 =	40	
Num =	90			
Den =	45			
K1 =	0.01	K2 =	0.01	
→ Eqm rho =	2	From closed-form solution		

rho	t
0.11	1
0.961111	2
1.915027	3
1.973232	4
1.991382	5
1.997208	6
1.999093	7
1.999705	8
1.999904	9
1.999969	10
1.999990	11
1.999997	12
1.999999	13
2.000000	14
2.000000	15
2.000000	16
2.000000	17
2.000000	18
2.000000	19
2.000000	20
2.000000	21
2.000000	22

t=1 → Initial prices:

				px =	1
				py =	9
				rho =	0.111111
Net x1 =	625	Net x2 =	140	Net X =	765
Net y1 =	-69.4444	Net y2 =	-15.5556	Net Y =	-85
				Mx =	7.65
				My =	0
				px+Mx =	8.65
				py+My =	9
				Sum =	17.65
				New px =	0.490085
				New py =	0.509915
		t=2 →		rho =	0.961111
Net x1 =	67.83237	Net x2 =	-19.1908	Net X =	48.64162
Net y1 =	-65.1944	Net y2 =	18.44444	Net Y =	-46.75
				Mx =	0.486416
				My =	0
				px+Mx =	0.976501
				py+My =	0.509915
				Sum =	1.486416
				New px =	0.65695
				New py =	0.34305
		t=3 →		rho =	1.915027
Net x1 =	31.553	Net x2 =	-29.5563	Net X =	1.996719
Net y1 =	-60.4249	Net y2 =	56.60109	Net Y =	-3.82377
				Mx =	0.019967
				My =	0
				px+Mx =	0.676917
				py+My =	0.34305
				Sum =	1.019967
				New px =	0.663666
				New py =	0.336334



← already close to $P_1 = 2/3$, $P_2 = 1/3$

$p = 1$