Residential Location

November 18, 2019

1 Residential location choice

In this section, we study a simple residential location model with homogeneous consumers. We will solve for a spatial equilibrium and numerically examine comparative statics with key parameters of the model.

1.1 Setup

An infinitely small business district is located on a featureless two-dimensional plane. The entire plane is owned by the profit maximizing landlord. The landlord rents land out to a highest bidder and does not derive any explicit utility from land. There is a population of farmers that value land at r_A per unit irrespective of the location. This means that price of land will be no less than r_A at any location.

The city is an area around the business district that is populated by a continuum of citizens. The size of the population is N. Each citizen maximizes utility $u(c, s)$, where c is the consumption of an aggregate good and s is the land consumption. The price of the consumption good is exogenous and is set to p. The price (rental rate) of land is denoted by r and set endogenously in equilibrium to clear the market for land. The land prices vary by location.

Each citizen collects income y by working at the business district. The citizen's budget constraint is $y - tx = pc + rs$, where x is the distance between citizen's home and business district and t is the cost of commute per distance traveled.

Since the plane is featureless and the citizen's budget constraint is affected by the distance between her home and the business district, two different locations that are characterized by the same x will feature the same prices. Thus, we conclude that, even though prices are allowed to vary by location, in equilibrium prices will depend only on the distance to the business district: $r(x)$. For the same reason, citizens' optimal choices will depend on x.

1.2 Spatial equilibrium

A spacial equilibrium is a collection of land prices $r(x)$, consumer choices $\{c_i^*, s_i^*, x_i^*\}_{i \in [0,N]}$ and landlord's choices of buyers of land at each location $i^* : \mathbb{R}_+ \to \emptyset \cup [0, N]$ such that:

- consumer choices of are optimal given prices;
- landlord rents out land to the highest bidder at each location: $i^*(x) \neq \emptyset \iff r(x) > r_A$;
- market for land clears for each location.

In order to find spatial equilibrium we need to follow these steps:

- Solve consumer maximization problem to find individual demands for land and consumption conditional on location x;
- Write a condition under which a citizen resides at location x: at each location that is populated in equilibrium, citizens must derive utility v^* which is independent of x. Each location that is not populated by citizens must induce a utility that is (weakly) lower than v^*. This condition restricts equilibrium prices $r(x)$ (usually via a differential equation).
- Identify locations that are populated by citizens using landlords profit maximization: every location in which $r(x)$ is strictly larger than r_A is populated by citizens.
- Find the remaining unknowns using market clearing condition.

1.3 Equilibrium characterization

We begin citizen's utility maximization problem. In order to find the individual demand for land at each location, we fix x and solve

$$\begin{aligned} &\max_{s,c}\{u(c,s)\} \\ &pc + r(x)s = y - tx \end{aligned}$$

The Lagrangian for this maximization problem is

$$\mathcal{L} = u(c,s) - \lambda(pc + r(x)s - y + tx)$$

Necessary conditions for the maximum are

$$\begin{aligned} u_1(c,s) &= \lambda p \\ u_2(c,s) &= \lambda r(x) \\ pc + r(x)s &= y - tx \end{aligned}$$

We can use one of the equations to solve for the Lagrange multiplier:

$$\begin{aligned} \frac{u_1(c,s)}{p} &= \frac{u_2(c,s)}{r(x)} \\ pc + r(x)s &= y - tx \end{aligned}$$

The first equation carries an important intuition: the citizen will optimally adjust his consumption of goods and land in such a way that marginal utility of consumption per dollar spent equals to the marginal utility of land per dollar spent. If one of the two commodities offers a higher marginal utility per dollar spent, the citizen can increase his wellbeing by putting more resources towards that commodity.

Let $c^*(p, r(x), y - tx)$ and $s^*(p, r(x), y - tx)$ be solutions to the maximization problem—these are individual demands for goods and land. If we plug these demands into the utility function, we obtain indirect utility of residing at location x given the prices $(p, r(x))$ and income y:

$$u(c^*(p, r(x), y - tx), s^*(p, r(x), y - tx) = v(x, p, r(x), y).$$

Suppose every location in $[0, x_B]$ is populated (x_B is a distance to the border of the city which is endogenously determined in equilibrium). Then,

$$v(x, p, r(x), y) = v^*, \forall x \leq x_B$$

or

$$\frac{dv(x, p, r(x), y)}{dx} = 0, \forall x \leq x_B.$$

By taking the derivative of v with respect to x and using citizen's optimality conditions we obtain (to save on notations we denote individual demands by c^* and s^* but keep in mind that these are function of prices, income and locations)

$$\begin{aligned} 0 &= \frac{du(c^*, s^*)}{dx} \\ &= \frac{du\left(\frac{y-tx-r(x)s^*}{p}, s^*\right)}{dx} \\ &= \frac{u_1(c^*, s^*)}{p}\left(-t - s^*\frac{dr(x)}{dx} - r(x)\frac{ds^*}{dx}\right) + \frac{u_2(c^*, s^*)}{r(x)}r(x)\frac{ds^*}{dx} \end{aligned}$$

From this equation we obtain an ordinary differential equation with respect to prices of land:

$$\frac{dr(x)}{dx} = -\frac{t}{s^*(p, r(x), y - tx)}.$$

This equation restricts prices in adjacent locations such that all locations are equally attractive. To see this intuition, note that indirect utility of residing at location x depends on the prices of land at that location. There is no other endogenous variable that affects the indirect utility.

This equation guarantees that $r(x)$ is a strictly decreasing function, therefore the region in which $r(x) \geq r_A$ is an interval $[0, x_B]$ as we conjectured previously. In particular, the boundary of the city is at distance x_B from the business district, where x_B solves $r(x_B) = r_A$. This gives us all necessary ingredient to state market clearing condition for land at each location.

Our approach to market clearing in this model is unusual: instead of allocating land to citizens, we will allocate citizens to land. Since we know individual demands for land, we can calculate exactly how many citizens we need to populate a plot of land of known size (i.e., we know what should be the density of population at each location). If we need exactly the mass N of citizens to populate a circular city with the border at distance x_B to the center, our markets for land clear. If we need more (less) than mass N of citizens, than we have excess supply (demand) of land.

First, note that the density of population at location x is

$$\frac{1}{s^*(p, r(x), y - tx)}.$$

Integrating the density of population multiplied by the area we obtain the total population in a city:

$$\int_0^{x_B} \frac{2\pi z}{s^*(p, r(z), y - tz)} dz = N.$$

This equation is the market clearing condition for land. The only unknown in this equation is x_B. The left-hand side of the equation is a continuous and strictly increasing function of x_B, therefore it has a unique solution. This concludes the characterization of the spacial equilibrium.

1.4 Free border-fixed border duality

In this section we examine how alternative assumptions on the mechanism that determines the border of the city can be imposed. In particular, consider a model in which instead of the featureless infinite plane, there is a disc with radius x_B with a business district in the center. In this alternative model, there are no farmers and citizens cannot rent land outside the disc. Thus, the border of the city is exogenously fixed.

We can use the solution for the model with the free border to find the solution for the model with a fixed border. If we find the value of r_A, such that the free border is exactly at the distance of x_B, the equilibrium prices in the two models will be exactly the same.

We can also solve the model with a free border assuming that the population of the city is not fixed at N but is subject to migration from the outside world. To do that, we assume that the city is a small open economy compared to the rest of the world and it does not affect the world prices. Suppose citizens achieve utility $\overline{u}$ by living outside the city (this is exogenous to the model by assumption that the city is small compared to the rest of the world). Population of the city will adjust in such a way that the citizen's utility from living in the city is also $\overline{u}$ (if the utility is below (above) this number, citizens will migrate from (to) the city). Therefore, in order to solve for an equilibrium, we need to find N such that the equilibrium utility of citizens is $\overline{u}$.

1.5 Key equations

Now we are ready to calculate spatial equilibrium numerically.

Let's start by reviewing the main equations that we will use to compute the equilibrium. First, the consumers' utility maximization problem is

$$\max_{s,c}\{u(c,s)\}$$
$$pc + r(x)s = y - tx$$

Since we are interested in individual demand for land, we can use budget constrain to exclude consumption from the problem:

$$s^*(r(x),p,y,x) = \arg\max_s \left\{ u\left(\frac{y - tx - r(x)s}{p}, s\right)\right\}$$

Note that we do not use first order conditions, but do the maximization numerically instead. An alternative approach would be to solve first order conditions numerically (you can try doing it yourself and compare the results).

The condition that the indirect utility must be independent of the location (otherwise, some locations will not be populated in equilibrium) gives us the equation that disciplines the difference in prices between two locations:

$$\frac{d}{dx}r(x) = -\frac{t}{s^*(p,r(x),y-tx)}$$

or in the integral form

$$r(x) - r(0) = -\int_0^x \frac{tdz}{s^*(p,r(z),y-tz)}$$

The endogenous city boundary x_B solves $r(x_B) = r_A$. The integral equation pins down prices at all locations except $x = 0$. In order to find the price in the center we solve the market clearing condition for land:

$$N = \int_0^{x_B} \frac{2\pi z dz}{s^*(p, r(z), y - tz)}$$

1.6 Algorithm

We will use the following algorithm for computing the equilibrium: 1. For given prices we compute the individual demand for land. 2. Using the integral equation we compute the land prices. 3. Iterate 1. and 2. until the prices that we use as input coincide with the prices that solve the integral equation. 4. Solve the market clearing condition for $r(0)$.

We cannot work with the continuum of locations directly, so we will intoduce a deterministic location grid with a small increment. The smaller is the increment, the better is the approximation, but the slower is the algorithm (you can play with the code to find the right trade-off).

1.7 Adding packages

We need to add some Julia packages that we use for our numerical implementation. These include `Optim`, `Plots` and `LaTeXStrings`.

```
In [2]: # using Pkg
        # Pkg.add("Optim")
        using Optim
        # using Pkg
        # Pkg.add("Plots")
        using Plots
        pyplot()
        # using Pkg
        # Pkg.add("LaTeXStrings")
        using LaTeXStrings
        #gr()
```

1.8 Implementation

We begin by defining consumer's preferences. In this example we will use Cobb-Duglas utility, but the script works for arbitrary utility functions (as long as the assumptions used in the model are satisfied).

```
In [41]: function utilityfn(c,s)
         #Utility function()
         alpha = 0.3
         rslt = (c.^alpha).*(s.^(1-alpha))
         end;
```

Now we can solve the utility maximization problem. Inputs for this problem are transportation costs t (variable `trc`), income y (variable `income`), the price of consumption good p (variable p), the discreet grid on the set of locations (variable xx) and the land prices (variable rr). Naturally, variables rr and xx should have the same dimensions. Function dmnd solves for demanded consumption and land for each location (so the outpus will be vectors with the same dimensions as xx) given inputs.

```
In [13]: function dmnd(trc, income, p, xx, rr)
         #demand
         nn = length(xx)
         s_temp = zeros(nn,1)
         c_temp = zeros(nn,1)
         for ii in 1:nn
             utilityfn_lambda(s) = -utilityfn(max(0,(income-trc*xx[ii]-rr[ii]*s))/p,s)
             s_temp[ii,1] = Optim.minimizer(optimize(utilityfn_lambda,0,max(0,(income-trc*xx[ii]
             c_temp[ii,1] = (income-trc*xx[ii]-rr[ii]*s_temp[ii,1])/p
         end
         s_dmnd = s_temp
         c_dmnd = c_temp
         return (c_dmnd, s_dmnd)
         end;
```

To test that our functions work we run a simple example with a very coarse grid:

```
In [14]: dmnd(1, 4, 1, [0.1; 0.2; 0.3], [0.3; 0.2; 0.1])

Out[14]: ([1.17; 1.14; 1.11], [9.1; 13.3; 25.9])
```

Once we have individual demand functions, we can solve the integral equation for land prices. We will iterate 1. and 2. until we arrive at the solution: this iterations are implemented in the while loop. Note that the prices in the iterations are updated with some inertia (see rr_old = (rr_old+rr_new)/2). This is a heuristic to obtain the convergence of the iterations. The function findrent below finds prices $r(x)$ given the price in the center $r(0)$ (we will solve for it later).

```
In [6]: function findrent(trc, income, p, r_A, xx, r0)
        #search for the solution to dr/dx = -t/s[r]

        toler1 = 0.001 #this is a precision parameter for iterations on r(x)
        nn = length(xx)
        drr = 12345.0
        rr_old = fill(r0, nn, 1)#r0*ones(length(xx),1)
        rr_new = rr_old.+drr


        while drr>toler1
            (c_dmnd, s_dmnd) = dmnd(trc, income, p, xx, rr_old)
            rr_new[1] = r0
            for ii in 2:nn
                if s_dmnd[ii] > 0
                    rr_new[ii] = max(rr_new[ii-1]-trc*(xx[ii]-xx[ii-1])/s_dmnd[ii] , 0)
                else
                    rr_new[ii] = 0
                end
            end
            drr = maximum(abs.(rr_new-rr_old), dims=1)[1]
            #println(drr)
            rr_old = (rr_old+rr_new)/2
        end

        return rr_new

                end;
```

To test that our functions work we run a simple example with a very coarse grid:

```
In [7]: findrent(0.1, 5, 2, 0.01, [1; 2; 3], 2)
```

```
Out[7]: 3×1 Array{Float64,2}:
         2.0
         1.9421856536472997
         1.8848740814169809
```

An alternative way to implement this fuction is to rely on the fact that $r(x)$ is continuous and use $r(x-\epsilon)$ with ϵ very small as an approximation for $r(x)$. This version of the function would work much faster. We will implement it later for the more computationally demanding model.

Finally, we can use market clearing condition to solve for $r(0)$. We use the binary search since we know that excess demand is monotone in $r(0)$. The first `while` look finds $r(0)$ large enough for the excess demand to be negative (recall, that this value is required to start the binary search). the second `while` loop solves the market clearing condition.

```
In [8]: function findeq(trc, income, p, r_A, xx, N)
        toler2 = 0.0001
        rr = similar(xx)
        dxx = similar(xx)
        dxx[1] = 0
        dxx[2:end] = xx[2:end]-xx[1:end-1]
        xxdxx = xx.*dxx



        r0h = 0.0 # this value must be low enough otherwise the solution will not be found
        r0l = 0.0
        rhs = 0.0 # arbitrary value less than N

        #Find the rough bound on r(0) that result in excess supply and excess demand for land
        while rhs[1]<N
            r0l = r0h
            r0h = 2*r0h+0.1
            rr = findrent(trc, income, p, r_A, xx, r0h)
            ii_B = argmin(abs.(rr.-r_A), dims=1)[1][1]
            (c_dmnd, s_dmnd) = dmnd(trc, income, p, xx, rr)
```

```
    rhs_mat = 2*pi*xxdxx./s_dmnd
    rhs = sum(rhs_mat[1:ii_B], dims=1)
    println("Right hand side is $rhs")
end

#r0l = 0
r0 = r0l
rhs = 0.0

while abs(r0h-r0l)>toler2
    if rhs[1]>N
        r0h = r0
    else()
        r0l = r0
    end
    r0 = (r0h+r0l)/2
    rr = findrent(trc, income, p, r_A, xx, r0)
    ii_B = argmin(abs.(rr.-r_A), dims=1)[1][1]
    (c_dmnd, s_dmnd) = dmnd(trc, income, p, xx, rr)
    rhs_mat = 2*pi*xxdxx./s_dmnd
    rhs = sum(rhs_mat[1:ii_B], dims=1)
    println("$r0l $r0h")
end

return rr

end;
```

Once all this functions are defined we can run them on some parameters. For instance, we can set

$$t = 1$$
$$y = 5$$
$$p = 1$$
$$r_A = 0.1$$
$$N = 20$$

and run the code to compute the prices.

```
In [27]: trc = 2.0
         income = 5.0
         p = 1.0
         r_A = 0.1
         N = 10.0
         grid_size = 300
         maxX = income/trc

         xx = [(maxX/grid_size)*(ii-1) for ii in (1:grid_size)]
```

```
        rr_bid = findeq(trc, income, p, r_A, xx, N)
        r_A_mat = r_A*ones(length(xx),1)
        rr = maximum([rr_bid r_A_mat], dims=2)
        ii_B = argmin(abs.(rr_bid.-r_A), dims=1)[1][1]
        border_x = xx[ii_B]

        (c_dmnd, s_dmnd) = dmnd(trc, income, p, xx, rr)

        V = utilityfn(c_dmnd[1],s_dmnd[1])

        dxx = copy(xx)
        dxx[1] = 0
        dxx[2:end] = xx[2:end]-xx[1:end-1]
        xxdxx = xx.*dxx

        WW = 2*pi*xxdxx.*(rr.-r_A)

        W = sum(WW[1:ii_B])

        println("====================================")

        println("utility:       $V")

        println("value of land: $W")
```

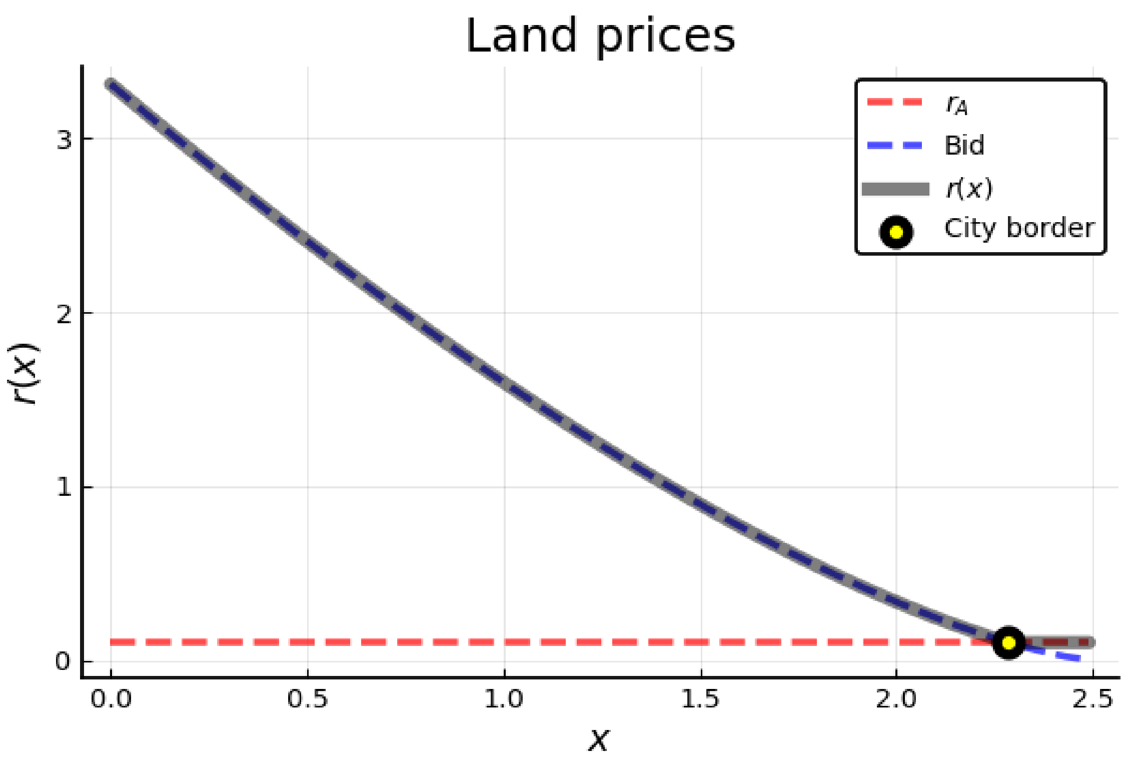
```
Right hand side is [0.0]
Right hand side is [0.401519]
Right hand side is [1.60213]
Right hand side is [4.15376]
Right hand side is [9.32504]
Right hand side is [19.7291]
3.1000000000000005 6.300000000000001
3.1000000000000005 4.700000000000001
3.1000000000000005 3.900000000000001
3.1000000000000005 3.500000000000001
3.3000000000000007 3.500000000000001
3.3000000000000007 3.400000000000001
3.3000000000000007 3.3500000000000005
3.3000000000000007 3.3250000000000006
3.3000000000000007 3.312500000000001
3.306250000000001 3.312500000000001
3.309375000000001 3.312500000000001
3.310937500000001 3.312500000000001
3.310937500000001 3.3117187500000007
3.310937500000001 3.311328125000001
3.310937500000001 3.3111328125000012
3.311035156250001 3.3111328125000012
```

```
====================================
utility:       1.1740750826605992
value of land: 13.865369684049831
```

Now we can use the output to plot a graph of $r(x)$.

```
In [28]: plot(xx, r_A_mat,
                 seriestype = :line,
                 linestyle = :dash,
                 linealpha = 0.7,
                 linewidth = 2,
                 linecolor = :red,
                 label = L"r_A")
         xlabel!(L"x")
         ylabel!(L"r(x)")
         title!("Land prices")
         plot!(xx, rr_bid,
                 seriestype = :line,
                 linestyle = :dash,
                 linealpha = 0.7,
                 linewidth = 2,
                 linecolor = :blue,
                 label = "Bid")
         plot!(xx, rr,
                 seriestype = :line,
                 linestyle = :solid,
                 linealpha = 0.5,
                 linewidth = 4,
                 linecolor = :black,
                 label = L"r(x)")
         scatter!([border_x], [r_A],
                 markershape = :circle,
                 markersize = 8,
                 markeralpha =1,
                 markercolor = :yellow,
                 markerstrokewidth = 3,
                 markerstrokealpha = 1,
                 markerstrokecolor = :black,
                 markerstrokestyle = :solid,
              #   dpi = 180,
             #    size = (1200,800),
                 thickness_scaling = 1.3,
                 label = "City border")
```

Out[28]:



```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-
```

1.9 Comparative statics

We have a code that computes spatial equilibrium. We can use it to do comparative statics with respect to parameters of interest.

1.9.1 Income

If income decreases, in the location, in which prices do not change, the new price $r_n(x)$ has a steeper slope than the old price $r_o(x)$:

$$r_n(x) = r_o(x) \implies \left|\frac{d}{dx} r_n(x)\right| > \left|\frac{d}{dx} r_o(x)\right|$$

This implies that, if such a location exists, the prices in the center increase and the prices close to the old border of the city decrease. It also implies that the equilibrium utility of the agents decreases.

We can now confirm our results using the numerical solutions for the equilibrium. We can also look at the effect on the total value of land. This effect is difficult to derive analytically, but easy to calculate numerically.

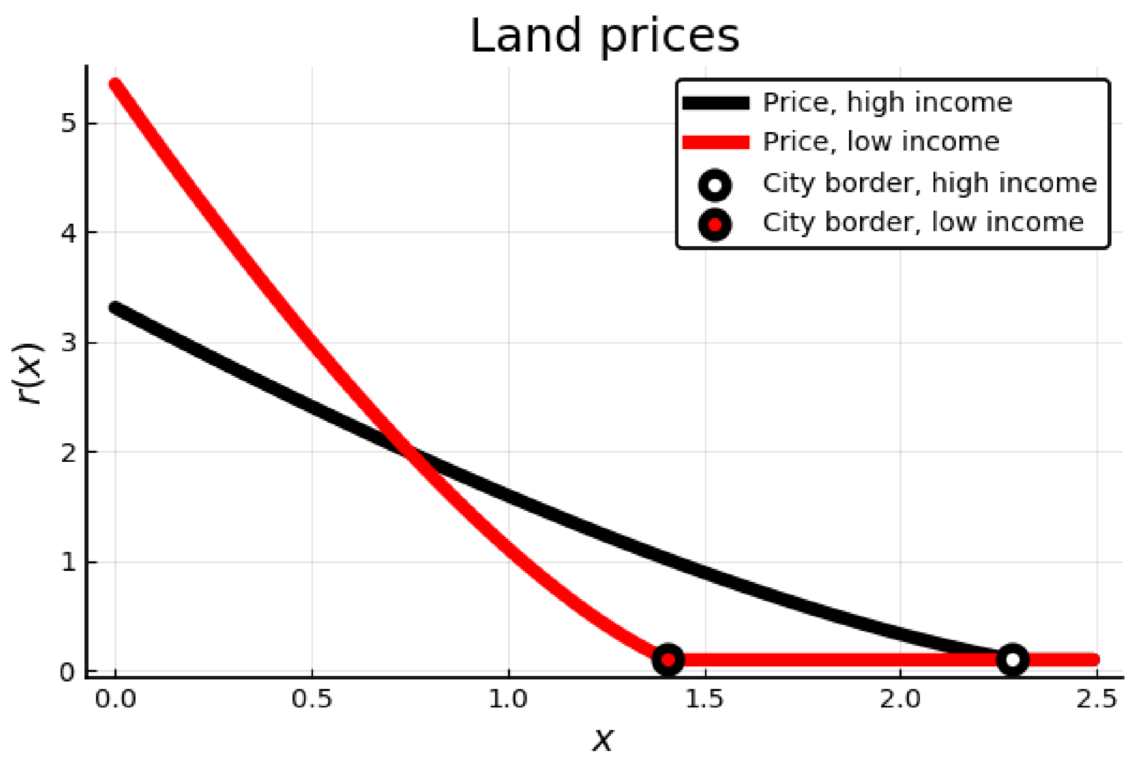
```
In [48]: income2 = 3.0 # in the previous computations, the income was 5.0
```

```
        rr2_bid = findeq(trc, income2, p, r_A, xx, N)

        rr2 = maximum([rr2_bid r_A_mat], dims=2)
        ii2_B = argmin(abs.(rr2_bid.-r_A), dims=1)[1][1]
        border2_x = xx[ii2_B]

        (c2_dmnd, s2_dmnd) = dmnd(trc, income2, p, xx, rr2)

        V2 = utilityfn(c2_dmnd[1],s2_dmnd[1])


        WW = 2*pi*xxdxx.*(rr2.-r_A)

        W2 = sum(WW[1:ii2_B])

        println("====================================")

        println("utility:       $V2")

        println("value of land: $W2")
```

```
Right hand side is [0.0]
Right hand side is [0.243616]
Right hand side is [0.966364]
Right hand side is [2.49925]
Right hand side is [5.61793]
Right hand side is [11.8444]
3.1000000000000005 6.300000000000001
4.700000000000001 6.300000000000001
4.700000000000001 5.500000000000001
5.100000000000001 5.500000000000001
5.300000000000001 5.500000000000001
5.300000000000001 5.4
5.300000000000001 5.3500000000000005
5.325000000000001 5.3500000000000005
5.3375 5.3500000000000005
5.34375 5.3500000000000005
5.346875000000001 5.3500000000000005
5.348437500000001 5.3500000000000005
5.348437500000001 5.34921875
5.348828125000001 5.34921875
5.348828125000001 5.3490234375000005
5.348828125000001 5.348925781250001
====================================
utility:       0.5035498071418909
value of land: 8.444621563957368
```

```
In [49]: plot(xx, rr,
                seriestype = :line,
                linestyle = :solid,
                linealpha = 1,
                linewidth = 4,
                linecolor = :black,
                label = "Price, high income")
        xlabel!(L"x")
        ylabel!(L"r(x)")
        title!("Land prices")
        plot!(xx, rr2,
                seriestype = :line,
                linestyle = :solid,
                linealpha = 1,
                linewidth = 4,
                linecolor = :red,
                label = "Price, low income")
        scatter!([border_x], [r_A],
                markershape = :circle,
                markersize = 8,
                markeralpha =1,
                markercolor = :white,
                markerstrokewidth = 3,
                markerstrokealpha = 1,
                markerstrokecolor = :black,
                markerstrokestyle = :solid,
             #   dpi = 180,
            #    size = (1200,800),
                thickness_scaling = 1.3,
                label = "City border, high income")
        scatter!([border2_x], [r_A],
                markershape = :circle,
                markersize = 8,
                markeralpha =1,
                markercolor = :red,
                markerstrokewidth = 3,
                markerstrokealpha = 1,
                markerstrokecolor = :black,
                markerstrokestyle = :solid,
             #   dpi = 180,
            #    size = (1200,800),
                thickness_scaling = 1.3,
                label = "City border, low income")
```

Out[49]:



```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-
```

1.9.2 Transportation costs

If transaction costs increase, the equilibrium utility of the citizen decreases.

Given that utility decreases, the prices in center of the city must increase because, the citizens who live in the center do not use transportation. The prices cannot increase everywhere because this will create an excess supply of land, so there exists a location in which the prices do not change.

In the location, in which prices do not change, the new price $r_n(x)$ has a steeper slope than the old price $r_o(x)$:

$$r_n(x) = r_o(x) \implies \left|\frac{d}{dx} r_n(x)\right| > \left|\frac{d}{dx} r_o(x)\right|$$

In order to complete the argument, we need to show that the equlibrium utility decreases with transportation costs. By contradiction, suppose that the utility increases (weakly). In that case, the new prices are below the old prices at each location and the new border of the city is closer to the center. A decrease in income net of transportation cost and an increase in utility implies that the demand for land increases at each location. However the total supply of land decreases because the border is supposed to move closer to the center. Therefore, there's a desired contradiction.

We can now confirm our results using the numerical solutions for the equilibrium. We can also look at the effect on the total value of land. This effect is difficult to derive analytically, but easy to calculate numerically.

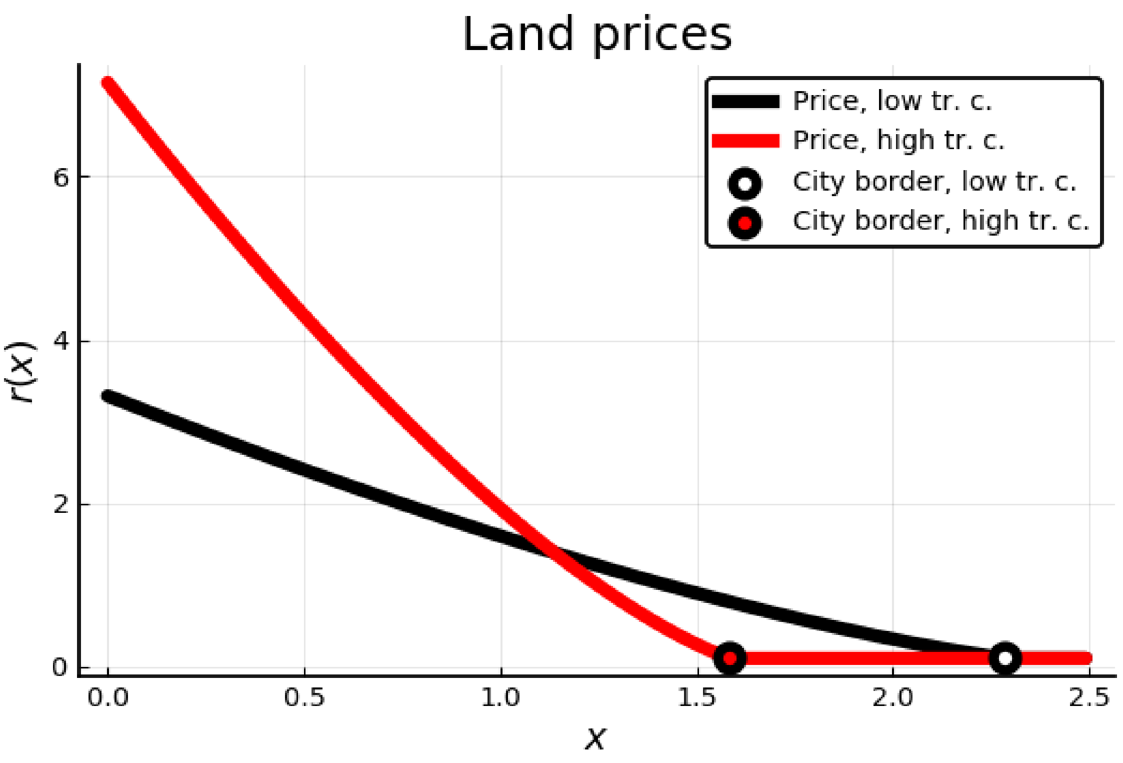
```
In [33]: trc3 = 3.0 # in the previous computations, the transportation costs per mile where 2.0

         rr3_bid = findeq(trc3, income, p, r_A, xx, N)

         rr3 = maximum([rr3_bid r_A_mat], dims=2)
         ii3_B = argmin(abs.(rr3_bid.-r_A), dims=1)[1][1]
         border3_x = xx[ii3_B]

         (c3_dmnd, s3_dmnd) = dmnd(trc3, income, p, xx, rr3)

         V3 = utilityfn(c3_dmnd[1],s3_dmnd[1])


         WW = 2*pi*xxdxx.*(rr3.-r_A)

         W3 = sum(WW[1:ii3_B])

         println("====================================")

         println("utility:       $V3")

         println("value of land: $W3")
```

```
Right hand side is [0.0]
Right hand side is [0.178022]
Right hand side is [0.716093]
Right hand side is [1.85]
Right hand side is [4.15297]
Right hand side is [8.77236]
Right hand side is [18.0071]
6.300000000000001 12.700000000000001
6.300000000000001 9.5
6.300000000000001 7.9
7.1000000000000005 7.9
7.1000000000000005 7.5
7.1000000000000005 7.300000000000001
7.1000000000000005 7.200000000000001
7.1000000000000005 7.15
7.125 7.15
7.1375 7.15
7.143750000000001 7.15
7.1468750000000005 7.15
7.1468750000000005 7.1484375
7.1468750000000005 7.147656250000001
```

```
7.1468750000000005 7.147265625000001
7.1468750000000005 7.1470703125
7.14697265625 7.1470703125
====================================
utility:       0.6851528166283354
value of land: 14.188929535518657
```

```
In [35]: plot(xx, rr,
                 seriestype = :line,
                 linestyle = :solid,
                 linealpha = 1,
                 linewidth = 4,
                 linecolor = :black,
                 label = "Price, low tr. c.")
         xlabel!(L"x")
         ylabel!(L"r(x)")
         title!("Land prices")
         plot!(xx, rr3,
                 seriestype = :line,
                 linestyle = :solid,
                 linealpha = 1,
                 linewidth = 4,
                 linecolor = :red,
                 label = "Price, high tr. c.")
         scatter!([border_x], [r_A],
                 markershape = :circle,
                 markersize = 8,
                 markeralpha =1,
                 markercolor = :white,
                 markerstrokewidth = 3,
                 markerstrokealpha = 1,
                 markerstrokecolor = :black,
                 markerstrokestyle = :solid,
              #   dpi = 180,
             #    size = (1200,800),
                 thickness_scaling = 1.3,
                 label = "City border, low tr. c.")
         scatter!([border3_x], [r_A],
                 markershape = :circle,
                 markersize = 8,
                 markeralpha =1,
                 markercolor = :red,
                 markerstrokewidth = 3,
                 markerstrokealpha = 1,
                 markerstrokecolor = :black,
                 markerstrokestyle = :solid,
              #   dpi = 180,
             #     size = (1200,800),
                 thickness_scaling = 1.3,
                 label = "City border, high tr. c.")
```

Out[35]:



```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-
```

1.9.3 Preference parameter α.

Similarly to the previous cases, one can show that, in the location, in which prices do not change, the new price $r_n(x)$ has a steeper slope than the old price $r_o(x)$:

$$r_n(x) = r_o(x) \implies \left|\frac{d}{dx}r_n(x)\right| > \left|\frac{d}{dx}r_o(x)\right|$$

This follows from the fact that the demand for land for a Cobb-Douglas utility function is

$$s^*(p, r(x), y - tx) = (1-\alpha)\frac{y - tx}{r(x)}.$$

Let's confirm this theoretical observation by running an example.

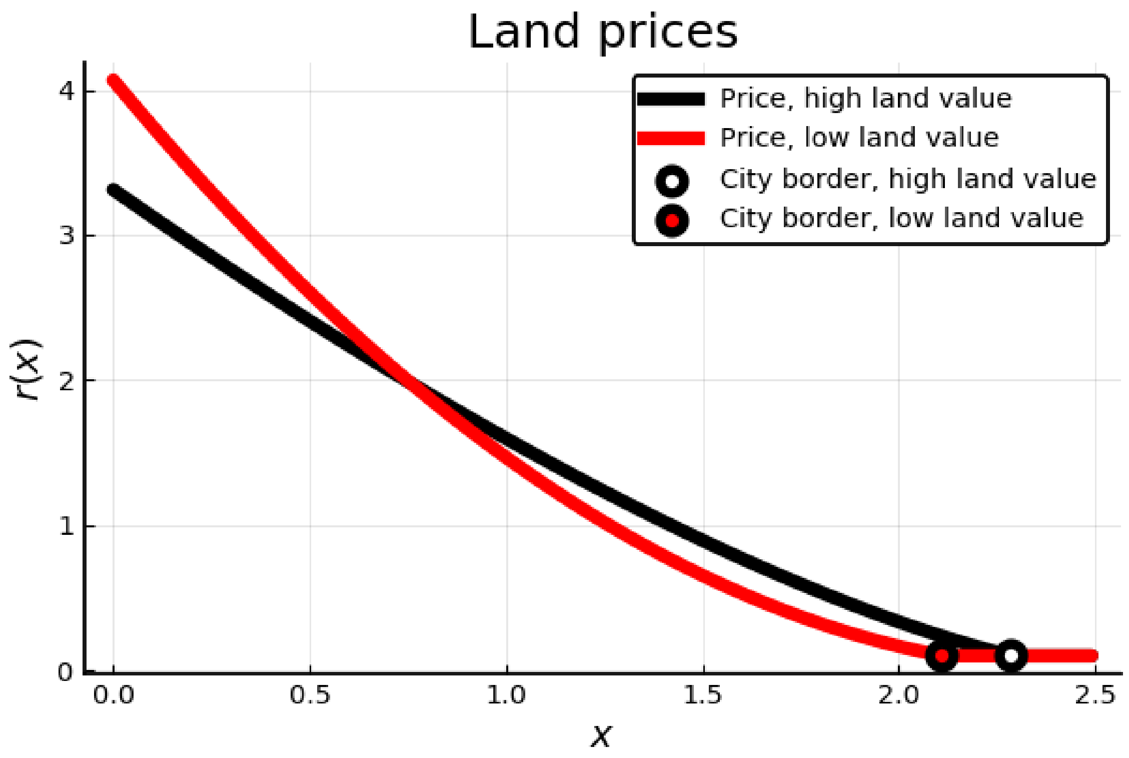
```
In [38]: function utilityfn(c,s)
         #Utility function()
         alpha = 0.5
         rslt = (c.^alpha).*(s.^(1-alpha))
             end; # in the previous computations, alpha was 0.3

         rr4_bid = findeq(trc, income, p, r_A, xx, N)

         rr4 = maximum([rr4_bid r_A_mat], dims=2)
         ii4_B = argmin(abs.(rr4_bid.-r_A), dims=1)[1][1]
         border4_x = xx[ii4_B]

         (c4_dmnd, s4_dmnd) = dmnd(trc, income, p, xx, rr4)

         V4 = utilityfn(c4_dmnd[1],s4_dmnd[1])


         WW = 2*pi*xxdxx.*(rr4.-r_A)

         W4 = sum(WW[1:ii4_B])

         println("====================================")

         println("utility:       $V3")

         println("value of land: $W3")
```

```
Right hand side is [0.0]
Right hand side is [0.305421]
Right hand side is [1.25567]
Right hand side is [3.30186]
Right hand side is [7.48763]
Right hand side is [15.8944]
3.1000000000000005 6.300000000000001
3.1000000000000005 4.700000000000001
3.900000000000001 4.700000000000001
3.900000000000001 4.300000000000001
3.900000000000001 4.1000000000000005
4.000000000000001 4.1000000000000005
4.050000000000001 4.1000000000000005
4.050000000000001 4.075000000000001
4.062500000000001 4.075000000000001
4.062500000000001 4.068750000000001
4.065625000000001 4.068750000000001
4.065625000000001 4.067187500000001
4.065625000000001 4.066406250000001
4.066015625 4.066406250000001
4.066015625 4.066210937500001
```

```
4.066015625 4.066113281250001
====================================
utility:       0.6851528166283354
value of land: 14.188929535518657
```

```
In [40]: plot(xx, rr,
                 seriestype = :line,
                 linestyle = :solid,
                 linealpha = 1,
                 linewidth = 4,
                 linecolor = :black,
                 label = "Price, high land value")
         xlabel!(L"x")
         ylabel!(L"r(x)")
         title!("Land prices")
         plot!(xx, rr4,
                 seriestype = :line,
                 linestyle = :solid,
                 linealpha = 1,
                 linewidth = 4,
                 linecolor = :red,
                 label = "Price, low land value")
         scatter!([border_x], [r_A],
                 markershape = :circle,
                 markersize = 8,
                 markeralpha =1,
                 markercolor = :white,
                 markerstrokewidth = 3,
                 markerstrokealpha = 1,
                 markerstrokecolor = :black,
                 markerstrokestyle = :solid,
              #   dpi = 180,
             #    size = (1200,800),
                 thickness_scaling = 1.3,
                 label = "City border, high land value")
         scatter!([border4_x], [r_A],
                 markershape = :circle,
                 markersize = 8,
                 markeralpha =1,
                 markercolor = :red,
                 markerstrokewidth = 3,
                 markerstrokealpha = 1,
                 markerstrokecolor = :black,
                 markerstrokestyle = :solid,
              #   dpi = 180,
             #    size = (1200,800),
                 thickness_scaling = 1.3,
                label = "City border, low land value")
```

Out[40]:



```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-
```