ECONOMICS BULLETIN

Fine Arts in Solow Model

Tetsuya Saito
Department of Economics, SUNY at Buffalo

Abstract

Applying some built-in functions of Mathematica, this note provides some graphics derived from convergent paths in "Solow Model." The main aim is to attarct other economists and introduces alternative analyzing tools but I also expect that this material helps teaching economics (programming codes are all shown in the appendix).

I am grateful to the anonymous referee and Dr John Conley, the editor of this journal, for some helpful comments to improve the initial version of this note and programming codes. All remaining errors and bugs in programming codes are mine.
Citation: Saito, Tetsuya, (2007) "Fine Arts in Solow Model." *Economics Bulletin,* Vol. 1, No. 3 pp. 1-20
Submitted: November 19, 2007. **Accepted:** December 15, 2007.
URL: http://economicsbulletin.vanderbilt.edu/2007/volume1/EB-07A10007A.pdf

1 Introduction

This paper is just for fun using one of the best known models in economics, "Solow Model (Solow 1957)," which is taught in almost all economics classes of all over the World. Applying functions (mainly `DensityPlot` and `3DPlot`) built in Mathematica, I draw pictures showing how many steps take to the steady state with respect to some combinations of parameters such as initial capital stock and saving rate. Then we find some fascinating pictured derived from this best known growth model. I also expect that this material helps teaching this exogenous growth model and some elementary dynamics. All programming codes are provided in the appendix.

2 Solow Model

Consider an economy producing and consuming a commodity Y. They produce Y using capital K labor L with a technology represented by a neo-classical production function $F : (L, K) \mapsto Y$ such that $F(0, K) = F(L, 0) = 0$ with Inada Condition. Because technological growth cannot be identified in simulations from other growth rates, I assume there is no technological change. Then consider motions of each factor $\dot{K}$ and $\dot{L}$:

$$\dot{L} = nL \quad \text{and} \quad \dot{K} = I - \delta K, \tag{1}$$

where I is the investment of this period, so that, it is $I = Y - C$ when C represents the consumption of this period. In Solow Model, C is exogenously given in terms of the saving rate $s \in [0, 1]$ as $C = sY$, so that, $I = (1 - s)Y$.

For simplicity of the analysis, as a convention, consider the model in terms of *per capita* variables that are denoted by corresponding lower case letters–*i.e.*, $y = Y/L$. Then, for the production function, we have

$$y = \frac{1}{L} \cdot F(K, L) = F(k, 1) := f(k). \tag{2}$$

For the motion of K, substituting $I = (1 - s)Y$, we have

$$\frac{\dot{K}}{L} = (1 - s)y - \delta k. \tag{3}$$

Now note

$$k = \frac{K}{L} \quad \Longrightarrow \quad \frac{\dot{k}}{k} = \frac{\dot{K}}{K} - \frac{\dot{L}}{L} = \frac{\dot{K}}{K} - n, \tag{4}$$

and then the left-hand side of (3) is given by

$$\frac{\dot{K}}{L} = \frac{\dot{K}}{K} \cdot \frac{K}{L} = \left(\frac{\dot{k}}{k} + n \right) \cdot k = \dot{k} + nk. \tag{5}$$

Therefore (3) is rewritten as

$$\dot{k} = (1-s)\cdot f(k) - (n+\delta)\cdot k. \tag{6}$$

If $\dot{k} = 0$, then we call it “steady state equilibrium” and denote such k by k^*.

3 Specification and Computing

Consider specifying the model for computing. Note $f'(k) > 0$ and $f''(k) < 0$, and still Inada Condition holds because

$$f'(k) = \frac{dF(K/L,1)}{dK}\frac{1}{dk/dK} = \frac{F_K(K,L)}{L}\cdot L = F_K(K,L) > 0, \tag{7}$$

and

$$f''(k) = \frac{dF_K(K,L)}{dK}\cdot\frac{1}{dk/dK} = L\cdot F_{KK}(K,L) < 0. \tag{8}$$

Noticing this, we have q well known and explicitly solvable candidate: $f(k) = k^\alpha$ for $\alpha \in (0,1)$ (notice, multiplying those functions by scalars just produces homeomorphic functions). For computation, because we cannot reach the steady state unless we start from the steady state, we need to clarify what is “convergence.”

Definition 1 *For sufficiently small $\varepsilon > 0$, we say the model have reached its steady state when $|k^* - k| \leqslant \varepsilon$.*

The simplest computable model is given by $f(k) = k^\alpha$ for $\alpha \in (0,1)$ and then k^* is given by

$$k^* = \left(\frac{1-s}{n+\delta}\right)^{\frac{1}{1-\alpha}} \tag{9}$$

Then consider an algorithm for computing. Let k_j be the capital stock at j-th stage of computing for $j = 0,1,2,\dots$, where k_0 is the initial capital stock and it is given. Starting from $k_0 > 0$, evaluate whether $|k^* - k_t|$ is larger than ε or not. While it returns “TRUE” then proceed to the next stage with accumulating capital stock in accordance with

$$k_{j+1} = k_j + \dot{k}_j = k_j + (1-s)\cdot f(k_j) - (n+\delta)\cdot k_j. \tag{10}$$

Once $|k^* - k_j| > \varepsilon$ turns to “FALSE,” then stop computing to return the total number of steps. Applying this algorithm to some combinations of parameters for each range, then we get density maps which take numbers of steps as their densities with respect to each combination of selected parameters. Appendix A provides the corresponding programming code.

4 Reviewing Graphics

In order to draw graphics, when they are given as parameters for each simulation, most cases applies $(s,\ \delta,\ n, k_0\ , \alpha,\ \varepsilon,\ \texttt{max}) = (0.4,\ 0.01,\ 0.01,\ 1 \text{ or } 10,\ 0.3,\ 0.00001,\ 5000)$, where `max` is the largest number of recursions when it does not converge (it avoids endless loops). However, for Figure 9 – Figure 13 apply parameters $(\delta,\ n,\ \varepsilon,\ \texttt{max}) = (0.1,\ 0.1,\ 0.001,\ 1000)$ instead of those of others because these computations take too long time. Results are visualized by `DensityPlot` which expresses numbers of steps to steady states as color gradations on the 2D surface for each combination of two parameters (each figure has the legend of colors).

Figure 1 shows the graphics derived by k_0 and n given other parameters. The thin curve represents locus of (n, k_0) such that $k_0 \simeq k^*$. Around that locus, he dynamics converge very fast comparing to other points and that is depicted by Figure 2 for the case $k_0 = 5$. It looks like a limestone cave. Figure 3 shows the 3D view below the surface (heights indicate respective steps to converge). Figure 4 depicts the relation between k_0 and s and it also have a canyon around the locus of (s, k_0) such that $k_0 \simeq k^*$ (the red diagonal area). As a 3D, Figure 5 shows its surface.

Next consider some cases k_0 is given. For the case $k_0 = 10$, Figure 6 depicts how combinations of s (horizontal) and δ (vertical) affect convergent paths. Because k_0 is sufficiently large, $k_0 \simeq k^*$ does not occur in this case. However, there is something on the figure that may be influenced by k^* (I don't know what is it). However, when $k_0 = 1$, we can identify the line $k_0 \simeq k^*$ as shown by Figure 7. Figure 8 shows influences of δ and n and that indicates number of steps are similar when $\delta \simeq n$. It also shows too low combinations bring too long way to the steady state but too high combinations does not indicate earlier convergence as well.

The most fascinating patterns appear when we see relations between input share α (vertical) and other parameters (horizontal). The first case is the relation with s and α when $k_0 = 1$ (Figure 9) and when $k_0 = 10$ (Figure 10). And the second case is the relation with δ and α when $k_0 = 1$ (Figure 11) and when $k_0 = 10$ (Figure 12). Those figures show we may encounter very long paths to converge in some applications that put high capital shares like $\alpha \geqslant 0.8$ (highly capital intensive society) around plausible s and δ. The last one (Figure 13) depicts the relation with k_0 and α that shows smaller α brings shorter paths than longer ones.

References

Solow, Robert (1957) 'Technical change and the aggregate production function.' *Journal of Economics and Statistics* 39(3), 312–320

Wolfram, Stephan (2003) *The Mathematica Book,* 5th ed. (Wolfram Media)

Appendix

A Programming Code

This is the source code for this paper.[1] I recommend readers who are willing to reproduce the same graphics to consider the size of mesh of the density map because large meshes consume computer resources and take too long to finish computing. In particular, commanding `PlotPoints -> {800, 800}` for `DensityPlot` and `PlotPoint -> {300, 300}` for `3DPlot` require over one hour to finish (1.42G PowerPC G4). It is reasonable to set `PlotPoints -> {100, 100}` initially and then increase them if something interesting patterns appear. In addition, `e` and `max` influence on the time of computing and I recommend readers not to increase those numbers larger than I give here unless higher spec machines are implemented.

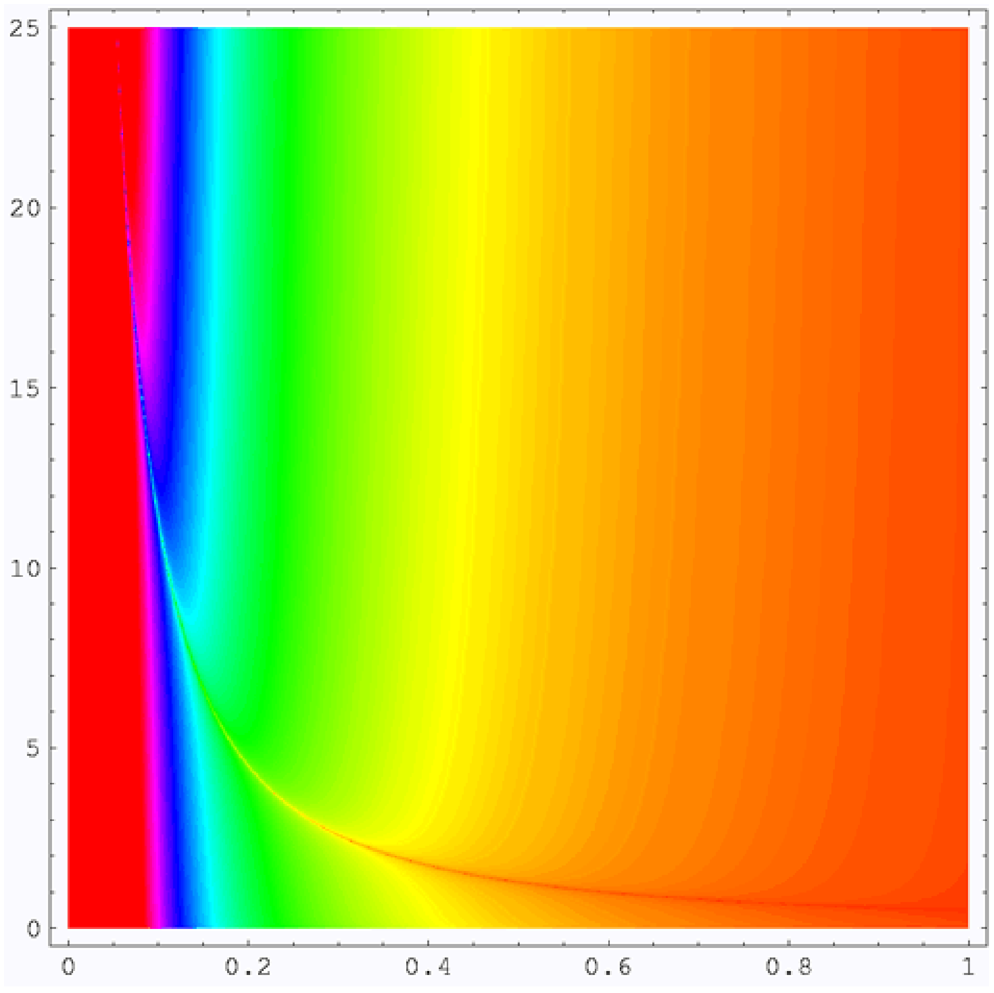
```
(* Definition of each variable *)
(*
  s : saving rate,
  n : population growth rate,
  d : capital depreciation rate,
  k0 : initial per - capita capital stock,
  a : parameter on production function,
  e : epsilon to determine convergence,
  max : maximum number of iterations
  *)

Solow[s_, n_, d_, k0_, a_, e_, max_] :=
  (
    (* Initializing parameters *)
    j = 0; (* Steps to the steady state *)
    k = k0; (* Initial capital stock *)
    b = 1/(1 - a); (* Just for convenience *)
    z = ((1 - s)/(n + d))^b; (* Steady state capital stock *)

    (* Loop while convergence *)
    While[
      Abs[z - k] > e && j < max, (* Evaluate whether close enough *)
      k = k + (1 - s)*k^a - (n + d)*k; (* Accumulate capital stock *)
      j++ (* Count each step *)
      ];

    (* Return total steps as the result *)
    Return[j];
    )

SolowLegend[T_] :=
  (
    max = Max[T];
    DensityPlot[
      x/max, {x, 0, max}, {y, 0, 1},
      PlotPoints -> {100, 100},
      Axes -> {True, False},
      Ticks -> Automatic,
      Mesh -> False,
      ColorFunction -> Hue,
      Frame -> False,
```

[1]See, for example, Wolfram (2003) for detailed explanations about each function.

```
      AspectRatio -> 0.05,
      TextStyle -> {FontSize -> 12}
      ]
    )

(* Draw pictures *)
(* Drawing initial capital vs depreciation rate *)
DensityPlot[
  Solow[0.4, x, 0.01, y, 0.3, 0.00001, 5000], {x, 0, 1}, {y, 0.001, 25},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
(*
  This process generates legend as follows;
  (1) Apply Solow function to get maximum steps;
  (2) Principally minimum is always zero--consider the case k0 = k*;
  (3) Then draw legend applying DensityPlot and Hue functions
  *)
T = Table[
      Solow[0.4, x, 0.01, y, 0.3, 0.00001, 5000], {x, 0, 1, 0.1}, {y, 0.001, 25, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Plot steps w.r.t. depreciation rate*)
ListPlot[
  Table[
    Solow[0.4, x, 0.01, 5, 0.3, 0.00001, 5000], {x, 0, 1, 0.0001}
    ]
  ]
(* Generate legend *)
T = Table[
      Solow[x, 0.1, 0.01, y, 0.3, 0.00001, 5000], {x, 0, 1, 0.1}, {y, 0.001, 25, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Drawing 2D for initial capital vs depreciation rate *)
ListPlot[
  Table[
    Solow[0.4, x, 0.01, 5, 0.3, 0.00001, 5000], {x, 0, 1, 0.0001}
    ]
  ]

(* Drawing 3D for initial capital vs depreciation rate *)
graph = Plot3D[
  Solow[0.4, x, 0.01, y, 0.3, 0.00001, 5000], {x, 0, 1}, {y, 0.001, 25},
  PlotPoints -> {300, 300}, Mesh -> False
  ]
(* Change viewpoint *)
Show[graph, ViewPoint -> {-1, -0.5, -0.5}, PlotRange -> {0, 300}]

(* Drawing initial capital vs saving rate *)
DensityPlot[
  Solow[x, 0.1, 0.01, y, 0.3, 0.00001, 5000], {x, 0, 1}, {y, 0.001, 25},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Flatten[T];
SolowLegend[T];
```

```
(* Drawing 3D for capital vs saving rate *)
Plot3D[
  Solow[x, 0.1, 0.01, y, 0.3, 0.00001, 5000], {x, 0, 1}, {y, 0.001, 25},
  PlotPoints -> {300, 300}, Mesh -> False
  ]

(* Drawing saving rate vs depreciation rate for k0=10 *)
DensityPlot[
  Solow[x, y, 0.01, 10, 0.3, 0.00001, 5000], {x, 0, 1}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[x, y, 0.01, 10, 0.3, 0.00001, 5000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Drawing saving rate vs depreciation rate for k0=1 *)
DensityPlot[
  Solow[x, y, 0.01, 1, 0.3, 0.00001, 5000], {x, 0, 1}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[x, y, 0.01, 1, 0.3, 0.00001, 5000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Drawing depreciation rate vs population growth *)
DensityPlot[
  Solow[0.4, x, y, 1, 0.3, 0.00001, 5000], {x, 0, 1}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[0.4, x, y, 1, 0.3, 0.00001, 5000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(*
 d and n are redefined as 0.1 for computational reason.
 By the same reason, e = 0.001 and max = 1000 are also redefined.
 *)

(* Drawing saving rate vs share of capital input for k0=1 *)
(* NOTE: Ignore some errors caused by dividing 0 *)
DensityPlot[
  Solow[x, 0.1, 0.1, 1, y, 0.001, 1000], {x, 0, 1}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[x, 0.1, 0.1, 1, y, 0.001, 1000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Drawing saving rate vs share of capital input for k0=10 *)
(* NOTE: Ignore some errors caused by dividing 0 *)
```

```
DensityPlot[
  Solow[x, 0.1, 0.1, 10, y, 0.001, 1000], {x, 0, 1}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[x, 0.1, 0.1, 10, y, 0.001, 1000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Drawing depreciation rate vs share of capital input for k0=1 *)
(* NOTE: Ignore some errors caused by dividing 0 *)
DensityPlot[
  Solow[0.4, x, 0.1, 1, y, 0.001, 1000], {x, 0, 1}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[0.4, x, 0.1, 1, y, 0.001, 1000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Drawing depreciation rate vs share of capital input for k0=10 *)
(* NOTE: Ignore some errors caused by dividing 0 *)
DensityPlot[
  Solow[0.4, x, 0.1, 10, y, 0.001, 1000], {x, 0, 1}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[0.4, x, 0.1, 10, y, 0.001, 1000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];

(* Drawing Initial capital vs share of capital input *)
(* NOTE: Ignore some errors caused by dividing 0 *)
DensityPlot[
  Solow[0.4, 0.1, 0.1, x, y, 0.001, 1000], {x, 0, 200}, {y, 0, 1},
  PlotPoints -> {800, 800}, Mesh -> False, ColorFunction -> Hue
  ]
(* Generate legend *)
T = Table[
      Solow[0.4, 0.1, 0.1, x, y, 0.001, 1000], {x, 0, 1, 0.1}, {y, 0, 1, 0.1}
      ];
T = Flatten[T];
SolowLegend[T];
```



Legend: Number of steps to the steady state

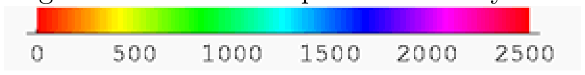


Figure 1: Initial capital stock vs population growth rate

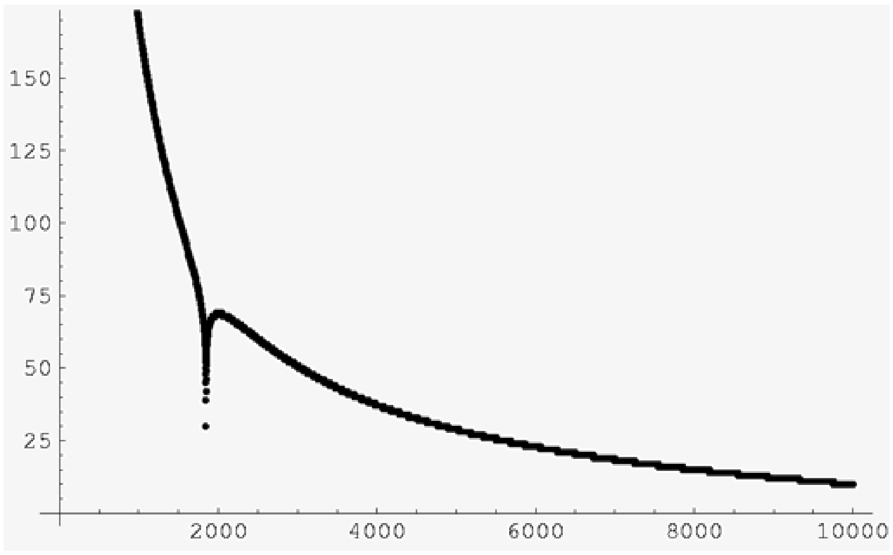


Figure 2: Initial capital stock vs population growth rate

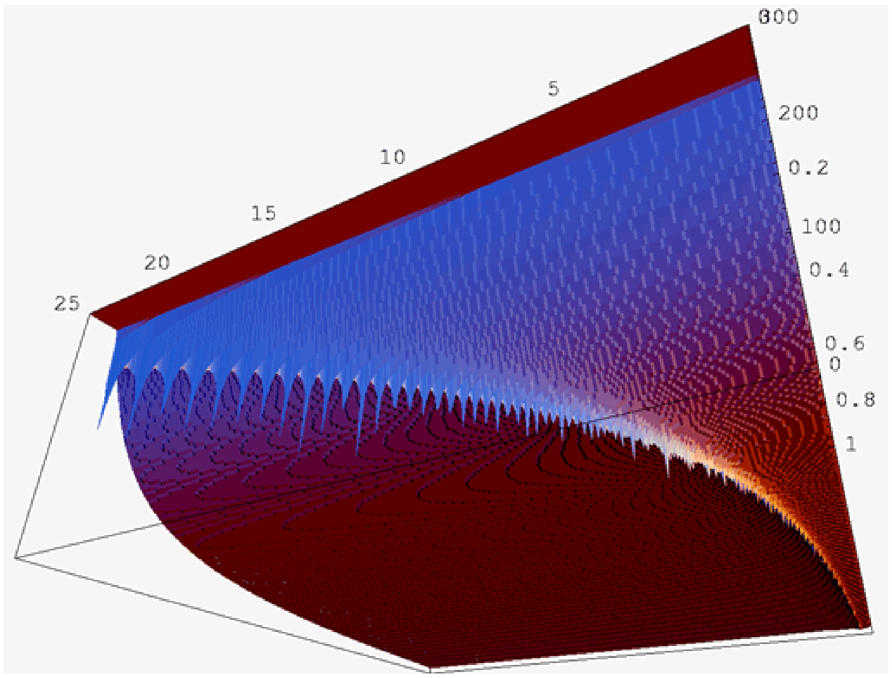


Figure 3: Initial capital stock vs population growth rate

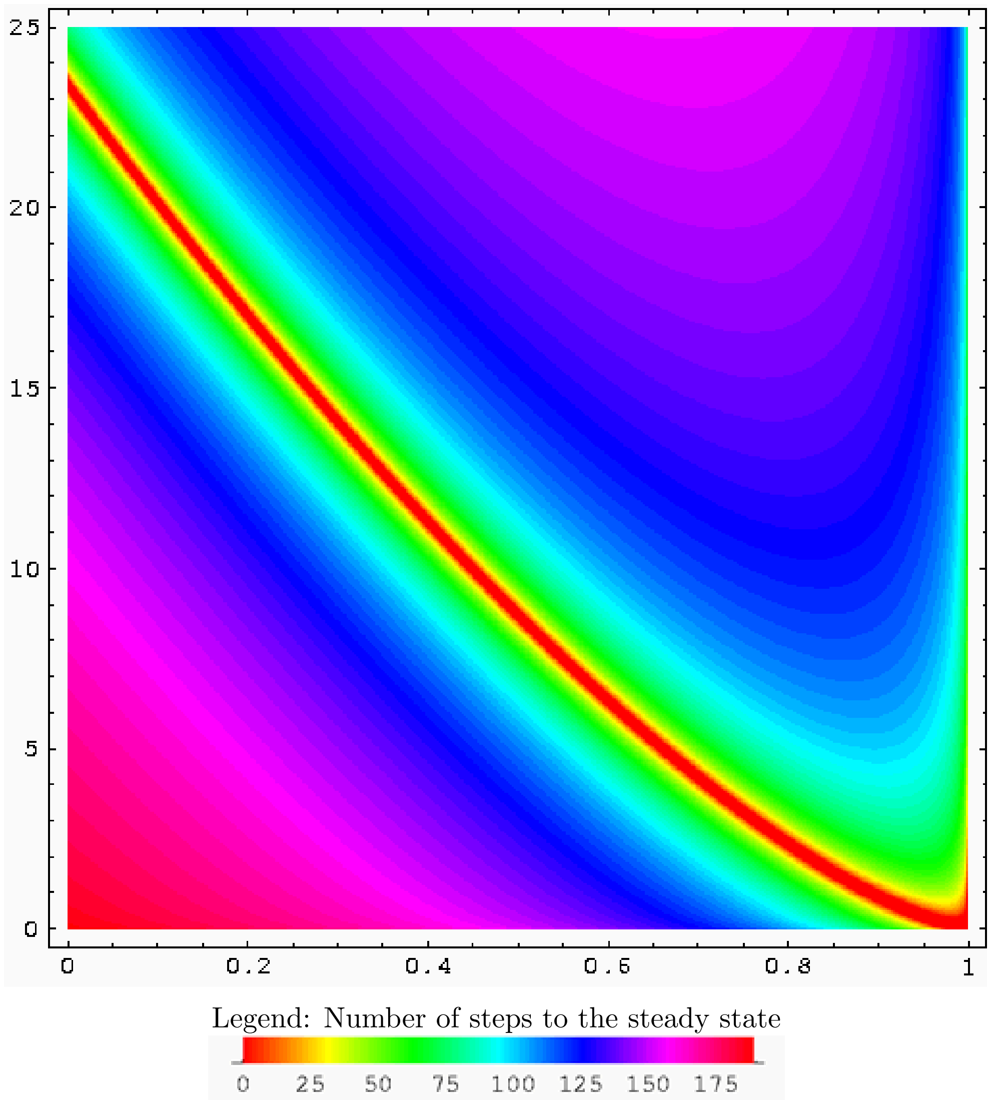


Figure 4: Initial capital stock vs saving rate

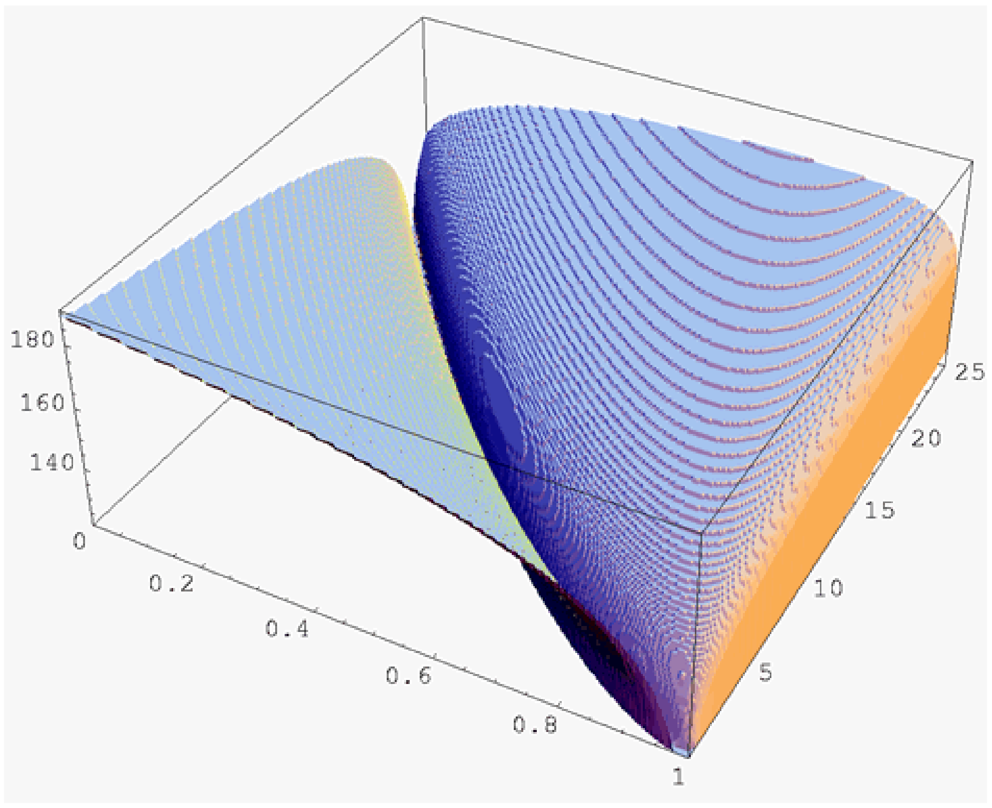
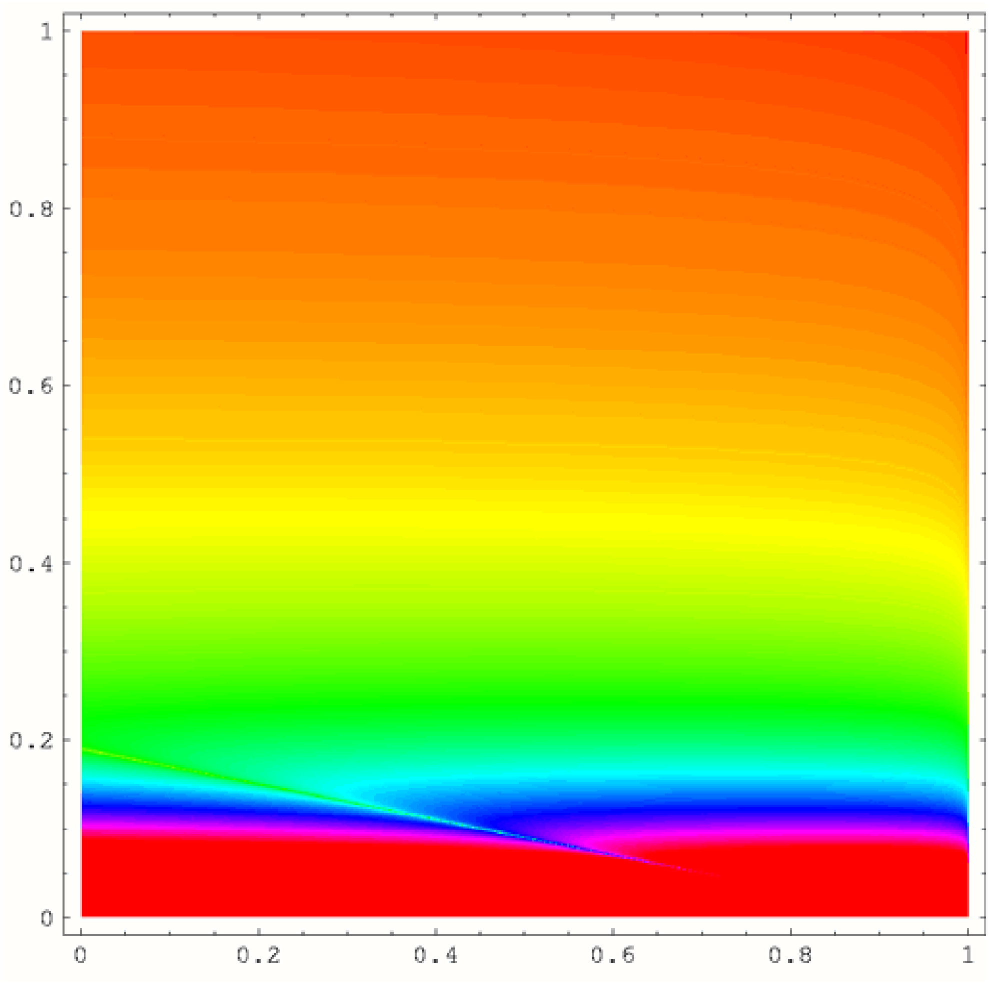


Figure 5: Initial capital stock vs saving rate



Legend: Number of steps to the steady state

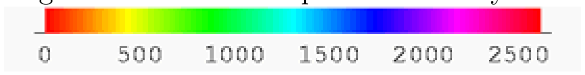
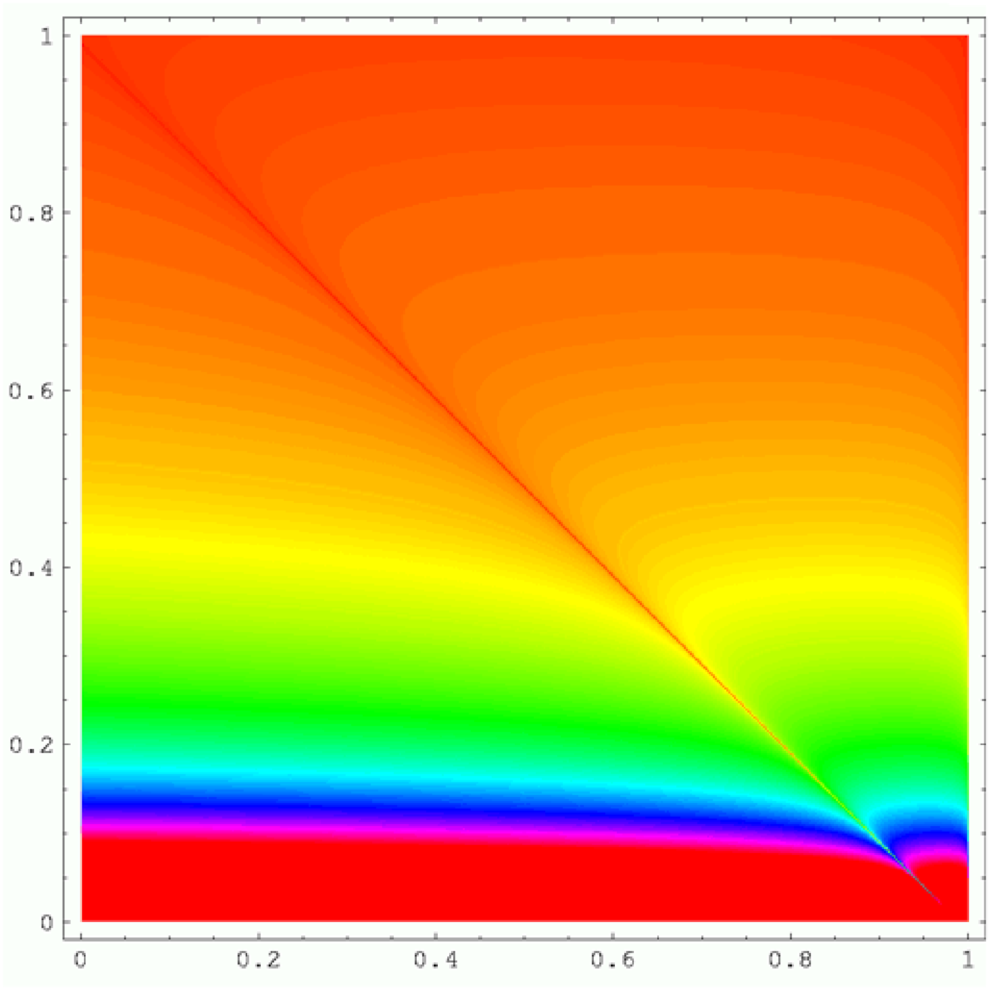


Figure 6: Saving rate vs depreciation rate ($k_0 = 10$)



Legend: Number of steps to the steady state

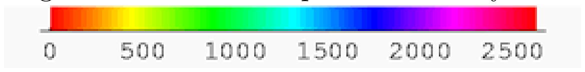
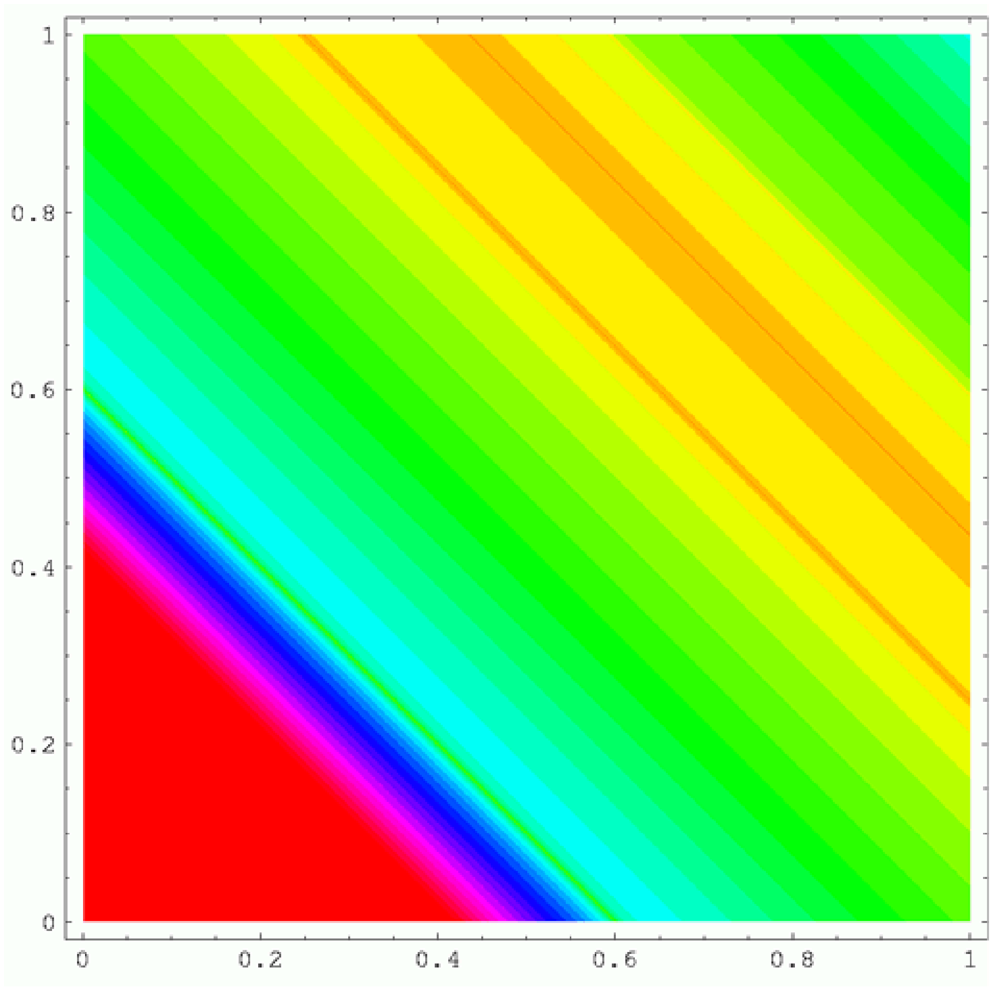


Figure 7: Saving rate vs depreciation rate ($k_0 = 1$)



Legend: Number of steps to the steady state

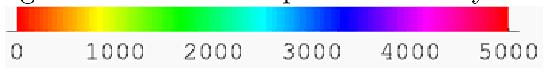


Figure 8: Depreciation rate vs population growth

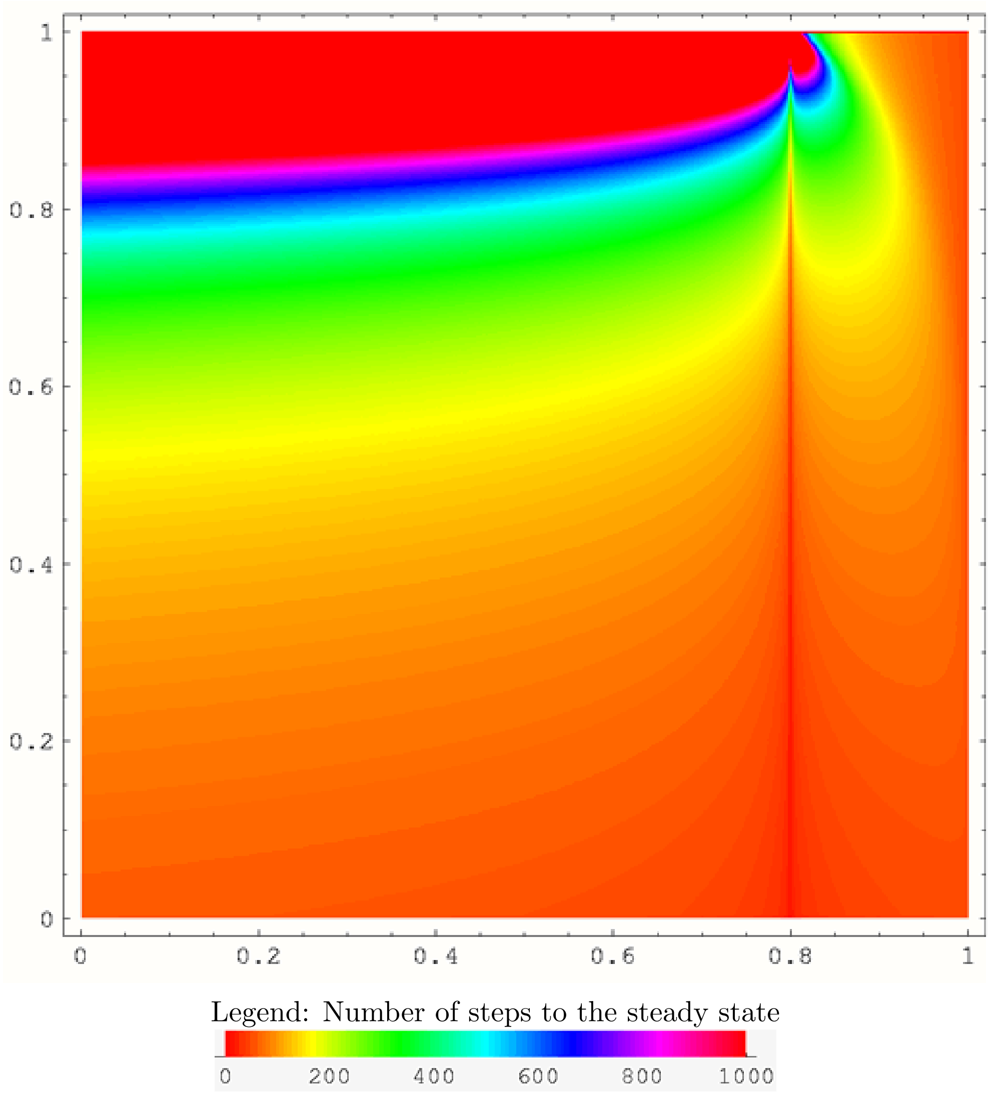
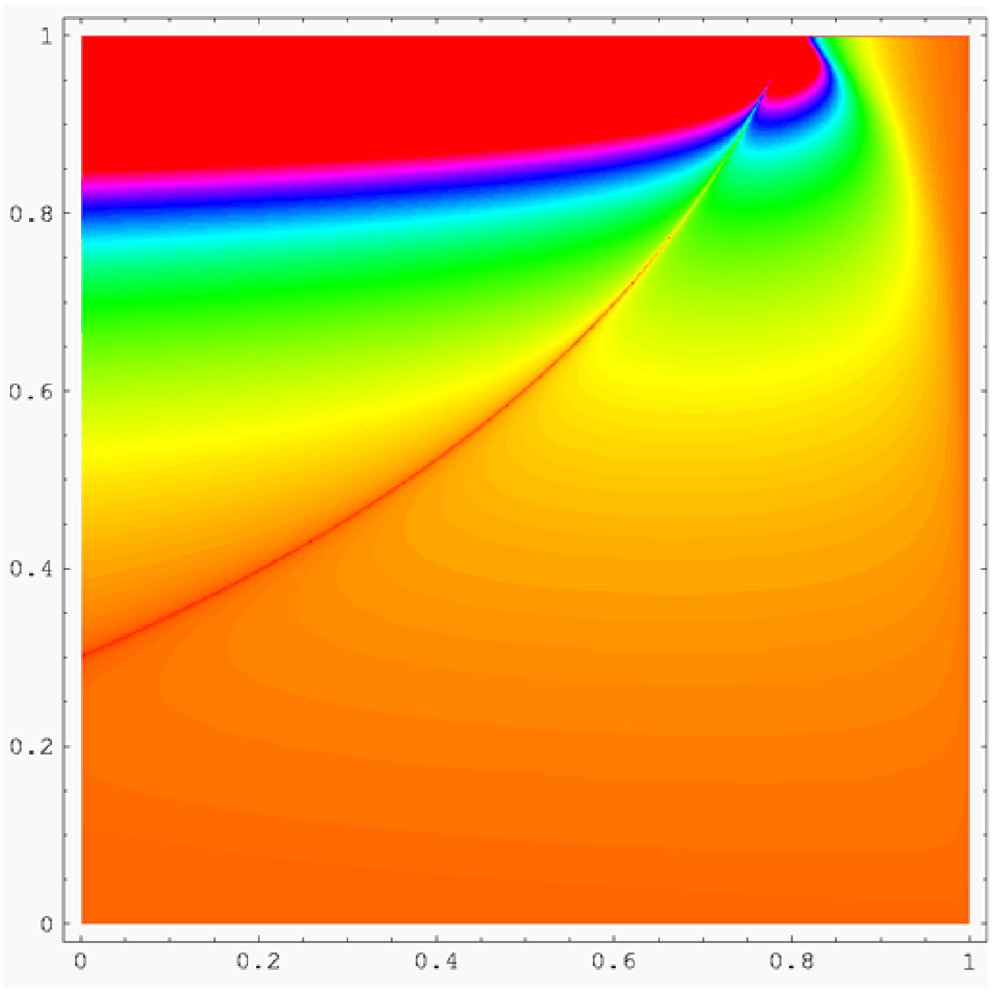


Figure 9: Saving rate vs share of capital input ($k_0 = 1$)



Legend: Number of steps to the steady state

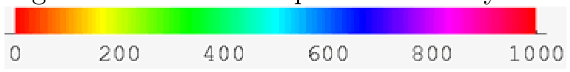
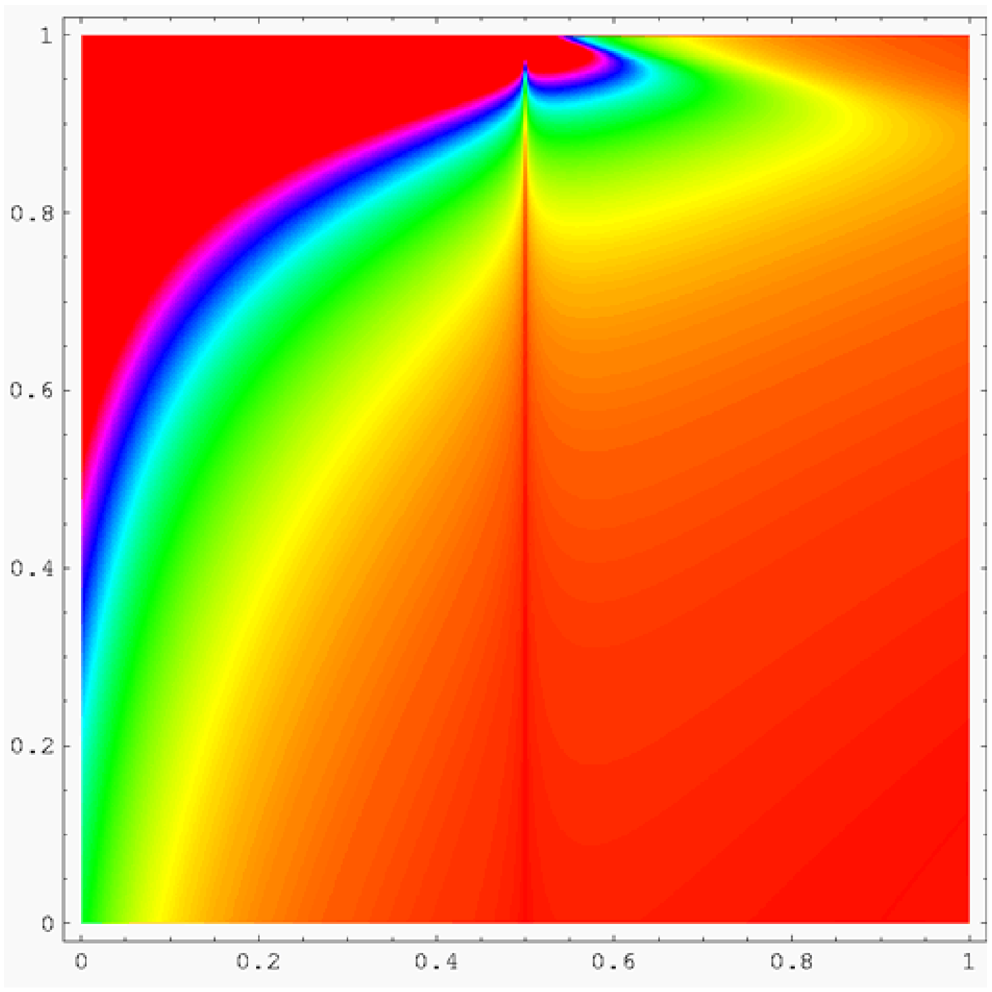


Figure 10: Saving rate vs share of capital input ($k_0 = 10$)



Legend: Number of steps to the steady state

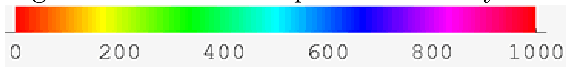


Figure 11: Depreciation rate vs share of capital input ($k_0 = 1$)

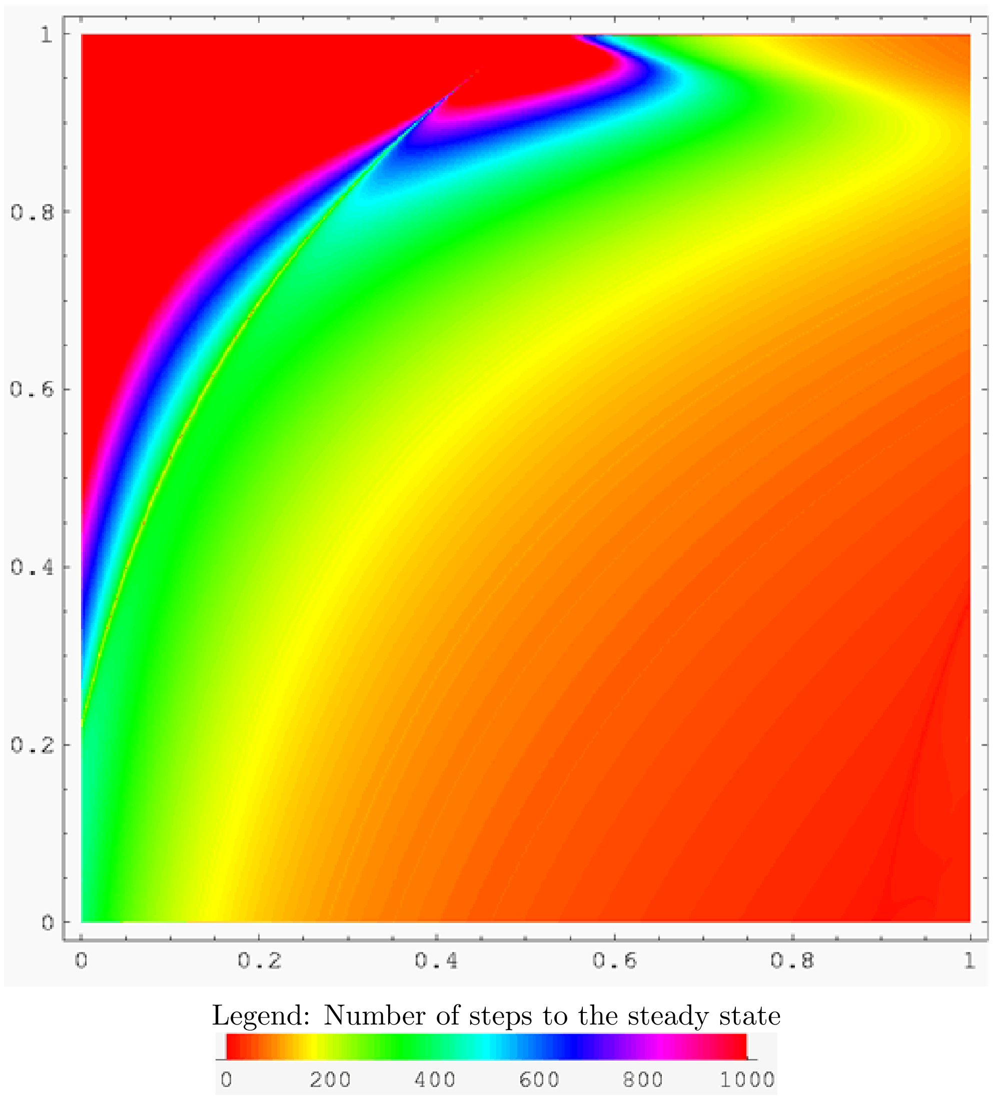


Figure 12: Depreciation rate vs share of capital input ($k_0 = 10$)

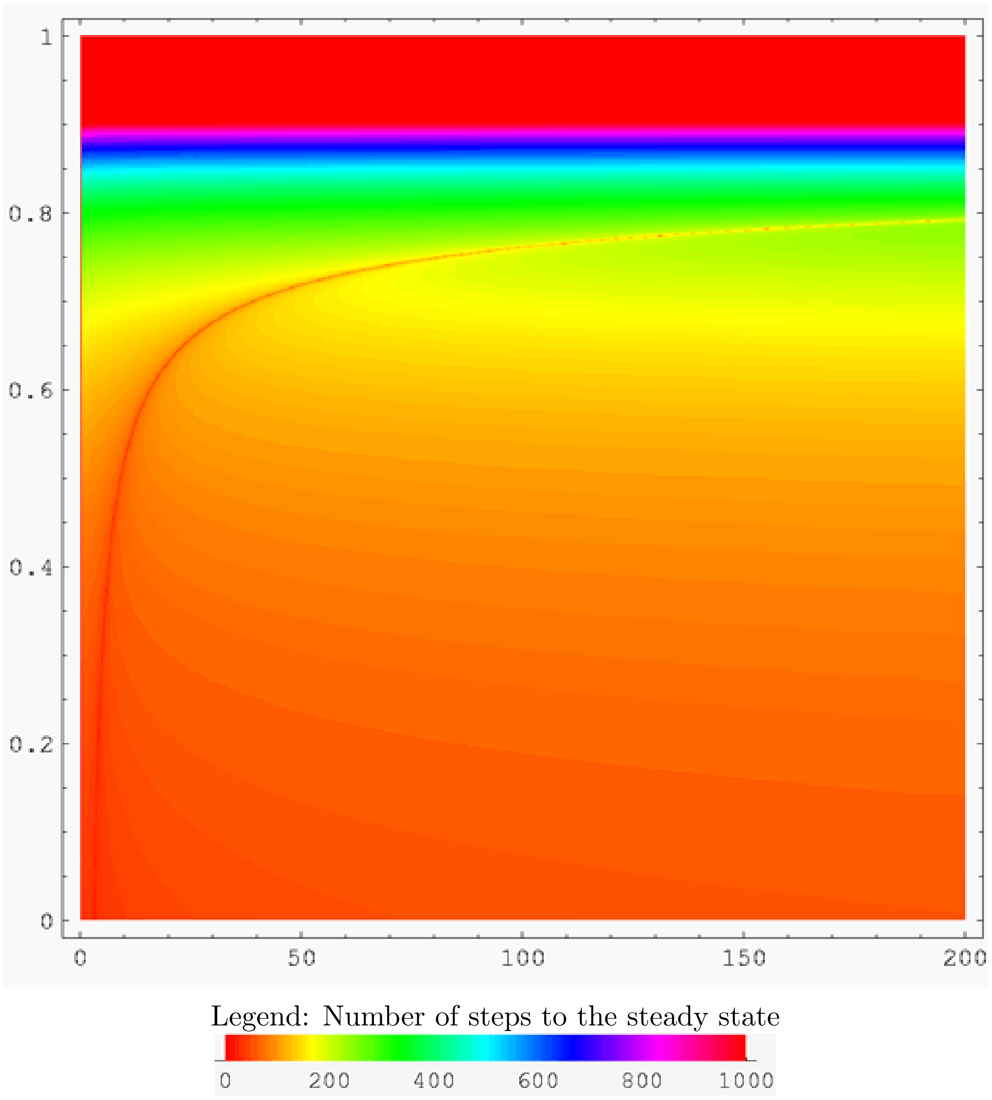


Figure 13: Initial capital vs share of capital input