Econ 712, Q2: Problem set 2 - Solution

Q1

(a)

Set up Social Planner's problem:

$$\max_{C_t\geq 0, N_t\in[0,1], K_{t+1}\geq 0} \sum \beta^t u(C_t, 1-N_t)$$

s.t. the resource constraint:

$$C_t + K_{t+1} - (1-\delta)K_t = F(K_t, A_t N_t)$$

Given K_0 and A_0. By taking the first order conditions and substituting out lagrange multiplier on the resource constraint we get:

Euler equation:

$$U_C(C_t, 1-N_t) = \beta U_C(C_{t+1}, 1-N_{t+1})[1-\delta + F_K(K_{t+1}, A_{t+1}N_{t+1})] \tag{1}$$

Consumption-labor supply tradeoff:

$$U_N(C_t, 1-N_t) = U_C(C_t, 1-N_t)A_t F_N(K_t, A_t N_t) \tag{2}$$

From the Firm's optimization problem:

$$r_t = F_K(K_t, A_t N_t) = F_K(\frac{K_t}{A_t N_t}, 1) \tag{3}$$

$$w_t = A_t F_N(K_t, A_t N_t) = A_t F_N(\frac{K_t}{A_t N_t}, 1) \tag{4}$$

Given K_0 and A_0 and (1)-(4), along with the resource constraint, we have 5 equations and 4 unknowns: C_0, N_0, w_0, r_0, with $C_{t+1} = (1+g)C_t$, $K_{t+1} = (1+g)K_t$, $Y_{t+1} = (1+g)Y_t$, $w_{t+1} = (1+g)w_t$, r_t is constant, N_t is constant. Just like in a model without growth, we need a specific K_0 to put us on the balanced growth path.

(b)

Assume $\gamma > 1$ (similar result holds for $\gamma = 1$). We guess and verify: Guess that C, Y, K, w grows at rate $(1+g)$, and that r, N is constant; then verify that this satisfies the equilibrium equations. Notice that we can rewrite (1) and (2) as

$$\Big(\frac{C_{t+1}}{C_t}\Big)^\gamma \frac{h(1-N_t)}{h(1-N_{t+1})} = \beta[1-\delta + F_K(\frac{K_{t+1}}{A_{t+1}N_{t+1}}, 1)] \tag{5}$$

$$\frac{h'(1-N_t)}{1-\gamma} C_t = h(1-N_t)A_t F_N(\frac{K_t}{A_t N_t}, 1) \tag{6}$$

On a balanced growth path,

$$(1+g)^{\gamma} = \beta[1-\delta+r_{t+1}] \tag{7}$$

$$\frac{h'(1-\bar{N})}{1-\gamma} C_t = h(1-\bar{N}) A_t F_N(\frac{K_t}{A_t \bar{N}}, 1) \tag{8}$$

$$K_{t+1}/K_t = (1-\delta) + Y_t/K_t - C_t/K_t \tag{9}$$

Notice that $\frac{K_t}{A_t N_t}$ is constant over time as long as N_t is constant at $\bar{N}$ and K_t grows at the same rate as A_t (which is $(1+g)$). Then, r_t is determined from (7) and is indeed a constant. Consumption grows at the same rate as A_t as seen from (8). Output $Y_t = A_t \bar{N} F(\frac{K_t}{A_t \bar{N}}, 1)$ and $w_t = A_t F_N(\frac{K_t}{A_t \bar{N}}, 1)$ grows at rate $(1+g)$ as well. (9) is satisfied as long as we choose a correct starting K_t.

(c)

We can rewrite everything in terms of units per efficient labor. Define $k_t = \frac{K_t}{A_t N_t}, c_t = \frac{C_t}{A_t N_t}, f(k_t) = F(k_t, 1)$. Then $r_t = f'(k_t)$, $w_t = A_t(f(k_t) - r_t k_t)$, $c_t + k_{t+1}(1+g) - (1-\delta)k_t = f(k_t)$. On the BGP $k_t = k_{t+1}$. Use (9) and resource constraint to analyze dynamics of $\frac{C_t}{A_t N_t}$ and $\frac{K_t}{A_t N_t}$,

$$(1+g) = \beta[1-\delta+f'(k_t)] \tag{10}$$

$$c_t + k_t(\delta+g) = f(k_t) \tag{11}$$

Notice that in response to an increase in the rate of depreciation δ both c_t and k_t fall on the BGP. Slope of the saddle path determines immediate change in c_t while there is no immediate change in k_t. What happens to C_t and K_t depend on response of N_t because $C_t = c_t A_t N_t$ and $K_t = k_t A_t N_t$. From equation (8),

$$\frac{h'(1-N_t)}{1-\gamma} N_t c_t = h(1-N_t) F_N(k_t, 1) \tag{12}$$

we cannot tell the direction of change in N_t.

(d)

Similar idea as before but now $N_t = 1$. Two equations, two unknowns:

$$(1+g)^{\gamma} = \beta[1-\delta+f'(k^{BG})] \tag{13}$$

$$c^{BG} + k^{BG}(\delta+g) = f(k^{BG}) \tag{14}$$

(e)

A fall in the rate of technological change shifts both locuses outwards (and to the right) so that c_t and k_t increase on the BGP (in the long run). Short run change in c_t depends on slope of the saddle path, and there is no change in k_t at the time of the change. Relatively flatter (steeper) saddle path leads to immediate increase (reduction) in c_t.

(f)

The fraction of output saved on the balanced growth path is $s_t = \frac{k_t(1+g)}{f(k_t)}$. Therefore, $\frac{\partial s_t}{\partial g} = \frac{k_t}{f(k_t)} + (g+1)k_g\left[\frac{1}{f(k_t)} - \frac{k_t f'(k_t)}{f^2(k_t)}\right]$. From (13), $k_g = \frac{(1+g)^{\gamma-1}\gamma}{\beta f''(k)}$. $\frac{\partial s_t}{\partial g} < 0$ implies $1 + (g+1)\frac{(1+g)^{\gamma-1}\gamma}{\beta f''(k_t)}\left[\frac{f(k_t)-k_t f'(k_t)}{k_t f(k_t)}\right] < 0$. For CD case $1 - \frac{(1+g)^{\gamma}\gamma}{(1+g)^{\gamma}-\beta(1-\delta)} < 0$ holds for $\gamma \geq 1$.

Q2

(a)

$$\max_{\{c_t\}_{t=0}^{\infty}} E_0 \sum_{t=0}^{\infty} \beta^t \frac{c_t^{1-\gamma}}{1-\gamma}$$

s.t. $0 \leq c_t \leq x_t$, $x_{t+1} = A_t(x_t - c_t)$, x_0 given.

Assume that c_t is chosen before A_t is realized. Here $\{c_t\}$ is a (feasible) sequence of functions of available information (think of them as contingency plans). If we define the product space $(Z^t, (2^Z)^t)$, where $Z = \{A_l, A_h\}$ and 2^Z is the power set of Z, and let $z^t \in Z^t$ be a partial history of shocks, then c_t is a measurable wrt z^{t-1}. The expection operator is taking expectations over $\lim_{t\to\infty}(Z^t, (2^Z)^t)$, that is, over the sequence of all shocks.

(b)

There are 2 potential states: Inherited savings, and realized shock. Since shocks are iid hence memoryless, all that matters is realized savings $A_{t-1}s_{t-1} = x_t$. The Bellman eq is

$$V(x) = \max_{s\in[0,x]}\{u(x-s) + \beta EV(As)\}$$

where $u(c) = \frac{c^{1-\gamma}}{1-\gamma}$ and $EV(As) = (1-\pi)V(A_l s) + \pi V(A_h s)$. To proceed to proving properties of the value function, we need to first handle the unboundedness of the utility function. Bounding the state space tough, since the production tech is linear and there is no depreciation: We may have to guess and verify (solve the whole problem first under some assumptions, then go back and check). But note that the utility function is homogenous of degree $0 < 1-\gamma < 1$. Along with the structure of the constraint set (a convex cone), this allows us to use similar arguments to the constant returns to scale case to deal with unboundedness (see handout 9). We will also get that V is homogenous of degree $1-\gamma$.

So under a specific norm and specific set of functions on X, denoted as $H(X)$, the operator $T : H \to H$ defined by $Tf(x) = \max_{s\in[0,x]}\{u(x-s) + \beta Ef(As)\}$ is a contraction mapping, with fixed point V. Continuity of V is a result of the theorem of the maximum, since $u(.)$ is continuous and our state space of shock is finite (Ef is continuous for f continuous).

V is increasing: Take any $x' \geq x$. Let $s^* \in \arg\max\{u(x-s) + \beta EV(As)\}$. Now $V(x') \geq u(x'-s^*) + \beta EV(As^*) \geq u(x-s^*) + \beta EV(As^*) = V(x)$.

V is concave: Take any concave $f \in H(X)$, and $(a,b) \in X^2$. Let $s_a \in \arg\max\{u(a-s) + \beta Ef(As)\}$, and define s_b correspondingly. Take any $\epsilon \in [0,1]$.

Let $x = \epsilon a + (1-\epsilon)b$, $s' = \epsilon s_a - (1-\epsilon)s_b$. Note that $s' \leq x$ Then

$$
\begin{aligned}
Tf(x) &\geq u(x - s') + \beta E f(As') \\
&= u(\epsilon(a - s_a) + (1-\epsilon)(b - s_b)) + \beta E f(As') \\
&\geq \epsilon u(a - s_a) + (1-\epsilon)u(b - s_b) \\
&+ \beta E[\epsilon f(As_a) + (1-\epsilon) f(As_b)] \\
&= \epsilon Tf(a) + (1-\epsilon)Tf(b)
\end{aligned}
$$

V is then concave by the collary to the contraction mapping theorem.

(c)

Since $u(.)$ is differentiable and our state space for shock is finite, differentiability of V follows from an application of the Benveniste-Sheinkman theorem. Taking FOCS:

$$-u'(x - s) + \beta E[AV'(As)] = 0$$

From the envelope conditions: $V'(x) = u'(x - s)$. Subbing back in gives

$$(x - s)^{-\gamma} = \beta E[A(As - s')^{-\gamma}]$$

Guess that $s(x) = \alpha x$. This gives $((1-\alpha)x)^{-\gamma} = \beta E[A((1-\alpha)\alpha x A)^{-\gamma}]$ $\iff 1 = \beta E[A^{1-\gamma}\alpha^{-\gamma}] \iff \alpha = (\beta E A^{1-\gamma})^{1/\gamma}$. Note that $\alpha < 1$ since $\beta A_h < 1$, so this confirms our guess.

From here, guess that $V(x) = Bx^{1-\gamma}$ for some B. This guess lends itself naturally from V being homogenous of degree $1-\gamma$. Then

$$
\begin{aligned}
Bx^{1-\gamma} = &\frac{((1-\alpha)x)^{1-\gamma}}{1-\gamma} + \pi B(A_h(1-\alpha)x)^{1-\gamma} \\
&+ (1-\pi)B(A_l(1-\alpha)x)^{1-\gamma}
\end{aligned}
$$

Then $B = \frac{(1-\alpha)^{1-\gamma}}{1-\gamma} + B(\pi(A_h(1-\alpha))^{1-\gamma} + (1-\pi)(A_l(1-\alpha))^{1-\gamma})$, which confirms our guess.

(d)

For the policy function to generate an optimal sequence, we apply Theorem 9.2 in Stokey, Lucas with Prescott. The main thing to show is that $\lim_{t\to\infty} E_t \beta^t V(A_{t-1}s_{t-1}) = 0$ (a sort of boundedness condition). Now $E_t V(A_{t-1}s_{t-1}) < V(A_h A_h^{t-1} x_0) = A_h^{t(1-\gamma)} V(x_0)$. Since $A_h \beta < 1$, $\lim_{t\to\infty} \beta^t A_h^{t(1-\gamma)} V(x_0) = 0$ so $\lim_{t\to\infty} E_t \beta^t V(A_{t-1}s_{t-1}) = 0$.

Q3

See attached codes. Comments are inlined. This was written in Julia, using Jupyter Notebooks, which I find very fast, flexible, and user friendly.

In [1]:
```
using Plots
```

In [15]:
```
### Params
β, δ, z, γ = 0.95, 0.1, 1.0, 2.0

### F
function F(k)
    return k^0.35
end

### u
function u(c, x)
    β, δ, z, γ = x
    return c^(1-γ) / (1-γ)
end
;
```

In [109]:
```
### Discretize grid of choice var (k)
kmin = 1e-3
kmax = δ^(-1/0.65) / 5
nk = 1000

kgrid = collect(range(kmin, kmax, length = nk))
;
```

In [79]:

```
### Bellman operator
function T(v, x, grid)
    β, δ, z, γ = x
    Tv = zero(v) ### Placeholder
    K = zero(grid) ### Placeholder for policy
    for (i, know) in enumerate(grid) ### Tv as function of k
        ynow = z * F(know) + (1-δ) * know
        val_next = zero(grid[grid .< ynow])
        for (ik, knext) in enumerate(grid[grid .< ynow]) ### Evaluating at each (feasibl
e) choice of knext
            val_next[ik] = u(ynow - knext, x) + β * v[ik]
        end
        res = argmax(val_next)
        Tv[i] = val_next[res]
        K[i] = grid[res]
    end
    return Tv, K
end

### Value function iteration
function VFI(vguess, x, grid; tol = 1e-4, maxiter = 1000, show = 1)
    β, δ, z, γ = x

    ### Initialize
    i = 0
    err = 1
    vnow = vguess
    K = zero(grid)

    while i < maxiter && err > tol ### Loop until 2 iterations are close enough, or max
 iteration is reached
        vnext, K = T(vnow, x, grid)
        err = maximum(abs.(vnext - vnow)) ### Supnorm
        i += 1

        vnow = vnext

        if i % 50 == 1 && show == 1 ### print some stuff so we dont get impatient waitin
g
            println("iter: ", i, " error: ", err)
        end
    end

    if i == maxiter
        println("maxiter reached")
    end

    return vnow, K
end
;
```

In [110]:
```
x = β, δ, z, γ
vguess = kgrid / (1-β) ### Arbitrary guess
V, K = VFI(vguess, x, kgrid)
;
```

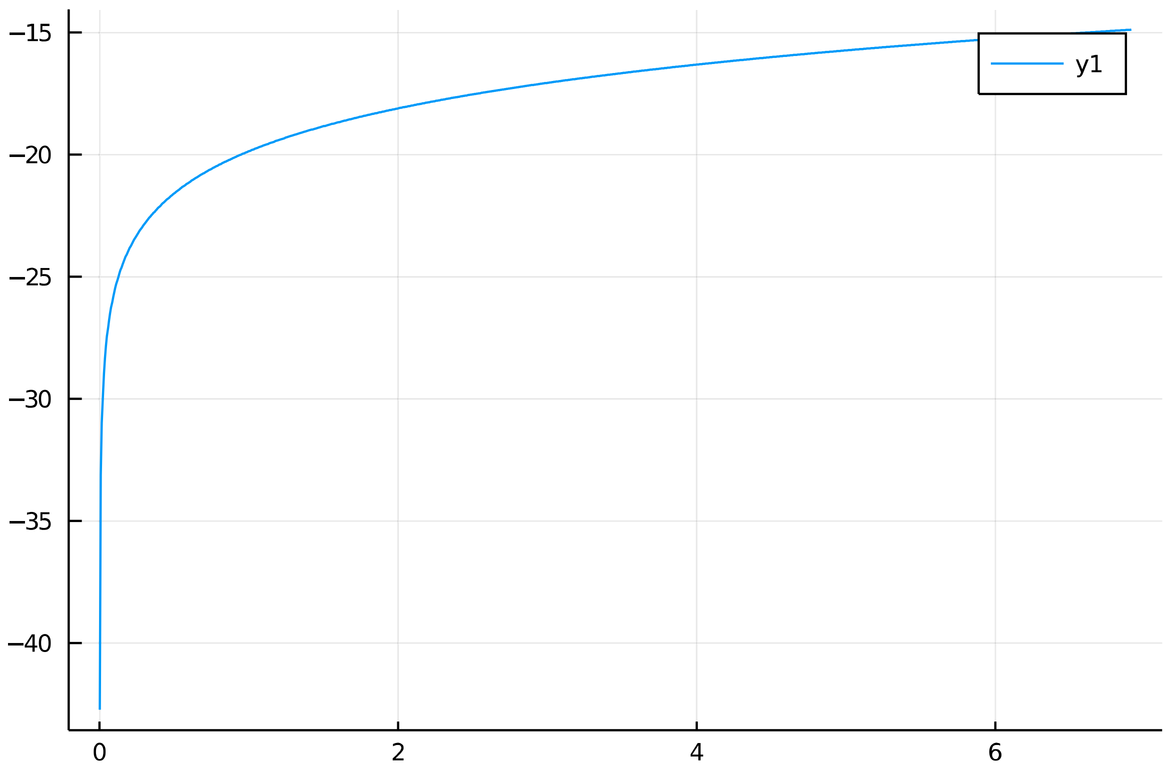
```
iter: 1 error: 11.233787938379756
iter: 51 error: 0.583823124907747
iter: 101 error: 0.04489735963609931
iter: 151 error: 0.0034546043439718233
iter: 201 error: 0.0002658144458465017
```

(a)

Value function and policy function

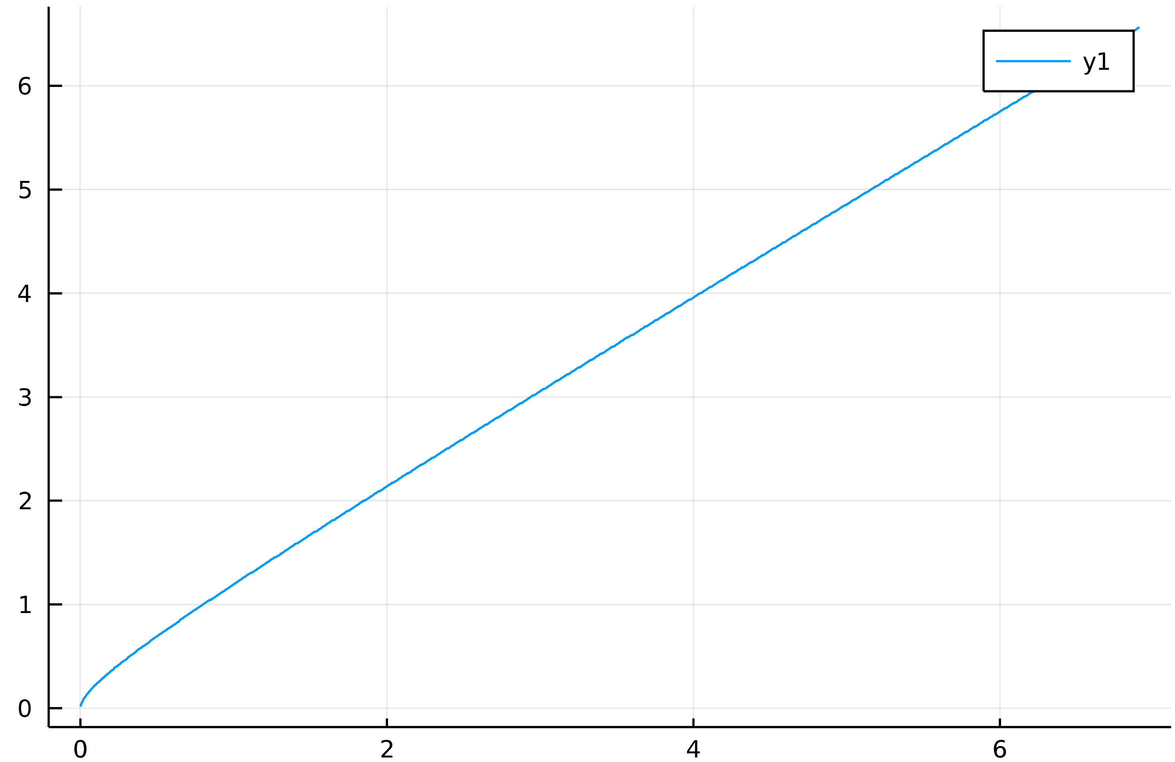
In [111]:
```
plot(kgrid, V)
```

Out[111]:



In [112]:
```
plot(kgrid, K)
```

Out[112]:



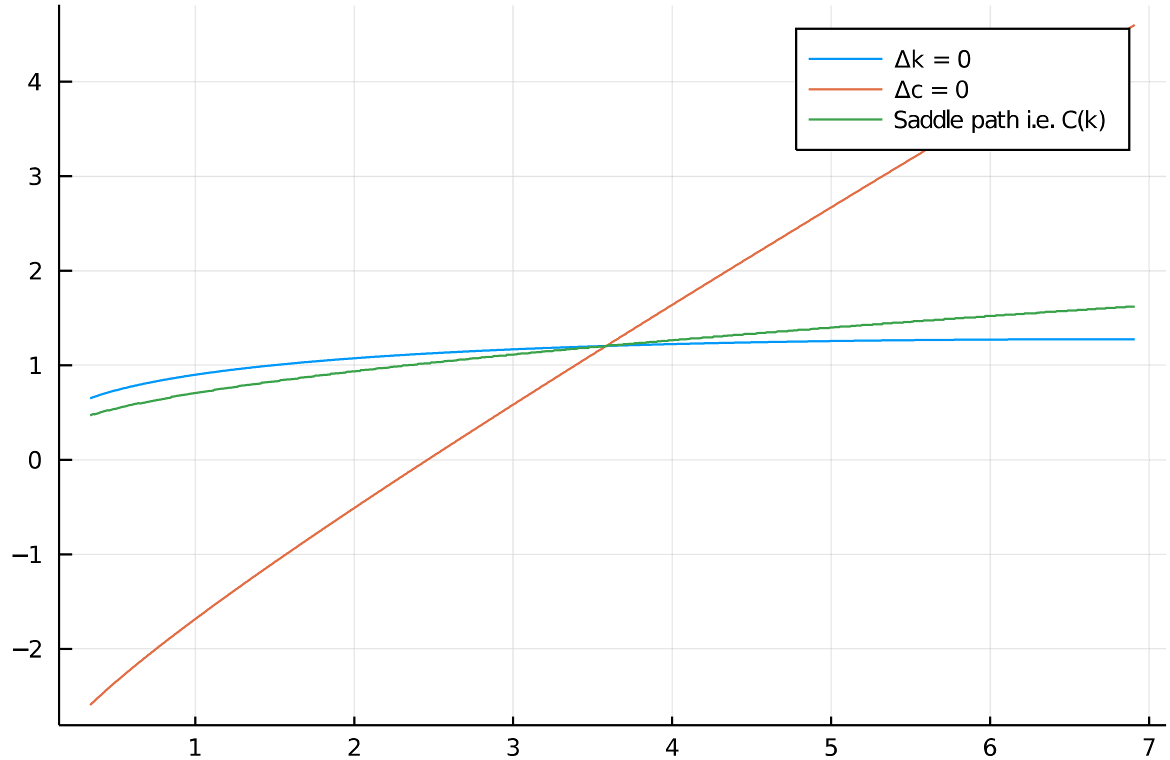
In [141]:
```
### Δk = 0
function get_line1(grid, x)
    β, δ, z, γ = x
    return z * F.(grid) - δ * grid
end

### Δc = 0
function get_line2(grid, x)
    β, δ, z, γ = x
    return z * F.(grid) + (1-δ) * grid .- ((1/β - 1 + δ) / 0.35 / z)^(-1/0.65)
end
;
```

Phase diagram

In [114]:
```
bcut = 50
fcut = nk
grid = kgrid[bcut:fcut]
plot(grid, get_line1(grid, x), label = "Δk = 0")
plot!(grid, get_line2(grid, x), label = "Δc = 0")
plot!(grid, z * F.(grid) + (1-δ) * grid - K[bcut:fcut], label = "Saddle path i.e. C(k)")
```

Out[114]:



(b)

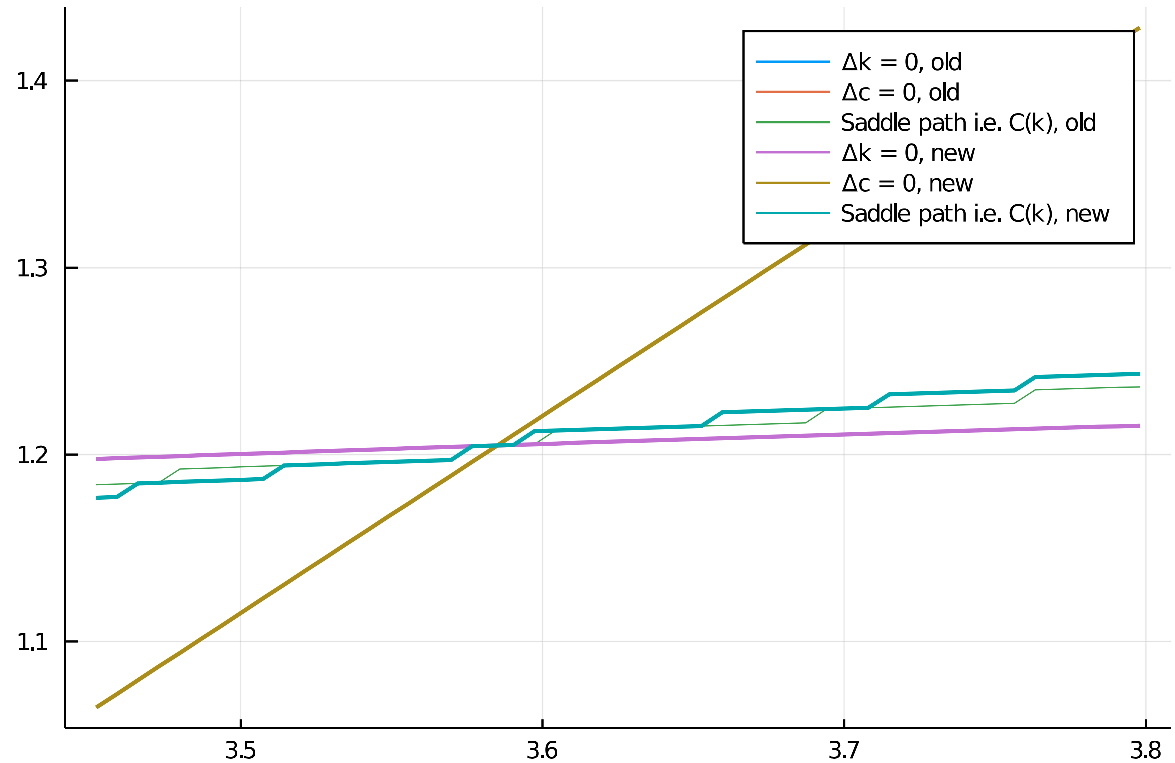
In [115]:
```
xb = β, δ, z, 1.01
vguess = kgrid / (1-β)
Vb, Kb = VFI(vguess, xb, kgrid)
;
```

```
iter: 1 error: 106.6667353691235
iter: 51 error: 8.198373564253188
iter: 101 error: 0.6308213212171268
iter: 151 error: 0.048538530965970494
iter: 201 error: 0.00373479606582805
iter: 251 error: 0.0002873737919344421
```

In [124]:
```julia
bcut = 500
fcut = 550
grid = kgrid[bcut:fcut]
plot(grid, get_line1(grid, x), label = "Δk = 0, old", lw = 0.5)
plot!(grid, get_line2(grid, x), label = "Δc = 0, old", lw = 0.5)
plot!(grid, z * F.(grid) + (1-δ) * grid - K[bcut:fcut], label = "Saddle path i.e. C(k),
 old", lw = 0.5)

plot!(grid, get_line1(grid, xb), label = "Δk = 0, new", lw = 2)
plot!(grid, get_line2(grid, xb), label = "Δc = 0, new", lw = 2)
plot!(grid, z * F.(grid) + (1-δ) * grid - Kb[bcut:fcut], label = "Saddle path i.e. C(k),
new", lw = 2)
```

Out[124]:



The steady state does not change, since γ does not enter into the steady state equations. The saddle path does though, it is more steep since there is more intertemporal substitution. Note the jaggyness of the saddle path, this is since we are discretizing the state space.

(c)

In [138]:
```julia
xb = β, δ, 1.2, γ
vguess = kgrid / (1-β)
Vb, Kb = VFI(vguess, xb, kgrid)
;
```

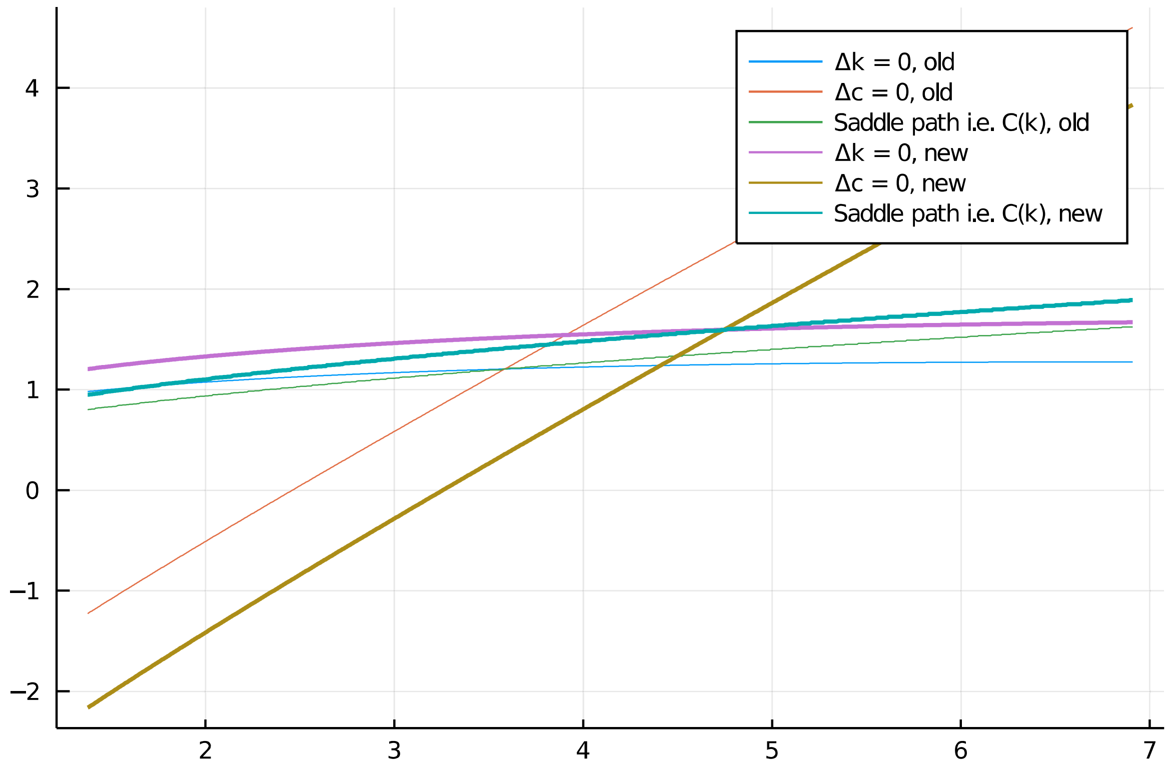
```
iter: 1 error: 16.841684435731352
iter: 51 error: 0.5760748952687885
iter: 101 error: 0.04432223124043766
iter: 151 error: 0.003410369388465284
iter: 201 error: 0.0002624107882862603
```

In [142]:

```
bcut = 200
fcut = nk
grid = kgrid[bcut:fcut]
plot(grid, get_line1(grid, x), label = "Δk = 0, old", lw = 0.5)
plot!(grid, get_line2(grid, x), label = "Δc = 0, old", lw = 0.5)
plot!(grid, z * F.(grid) + (1-δ) * grid - K[bcut:fcut], label = "Saddle path i.e. C(k),
 old", lw = 0.5)

plot!(grid, get_line1(grid, xb), label = "Δk = 0, new", lw = 2)
plot!(grid, get_line2(grid, xb), label = "Δc = 0, new", lw = 2)
plot!(grid, 1.2 * F.(grid) + (1-δ) * grid - Kb[bcut:fcut], label = "Saddle path i.e. C
(k), new", lw = 2)
```

Out[142]:



With higher z, we get higher consumption and capital in the steady state. With $\gamma = 2$ sufficiently high here, we have that consumption will jump up to the new saddle path, afterwards consumption and capital will increase to their new steady state levels.

In [ ]: