Spencer Lyon

Sargent - RA: RMT4 P 2.1 - 2.5

Due on Sunday, July 14, 2013

July 20, 2013

Problem 2.1

Consider the Markov Chain $(P,\pi_0) = \left(\left[\begin{smallmatrix}.9 & .1\\ .3 & .7\end{smallmatrix}\right], \left[\begin{smallmatrix}.5\\ .5\end{smallmatrix}\right]\right)$ and a random variable $y_t = \bar{y}x_t$, where $\bar{y} = \left[\begin{smallmatrix}1\\ 5\end{smallmatrix}\right]$. Compute the likelihood of the following three histories for y_t for $t = 0, 1 \ldots 4$:

a. $1,5,1,5,1$

b. $1,1,1,1,1$

c. $5,5,5,5,5$

Each part in this problem is fairly straight-forward. I simply need to pick the correct element of P and multiply them all together with the correct starting value of π_0.

a. $P_{21}P_{12}P_{21}P_{12}\pi_{01} = (.3)(.1)(.3)(.1)(.5) = 0.00045$

b. $P_{11}P_{11}P_{11}P_{11}\pi_{01} = (.9)(.9)(.9)(.9)(.5) = 0.32805$

c. $P_{22}P_{22}P_{22}P_{22}\pi_{02} = (.7)(.7)(.7)(.7)(.5) = 0.12005$

□

Problem 2.2

Consider a two-state Markov chain. Consider a random variable $y_t = \bar{y}x_t$ where $\bar{y} = \left[\begin{smallmatrix}1\\ 5\end{smallmatrix}\right]$. It is known that $E(y_{t+1}|x_t) = \left[\begin{smallmatrix}1.8\\ 3.4\end{smallmatrix}\right]$ and that $E(Y_{t+1}^2|x_t) = \left[\begin{smallmatrix}5.8\\ 5.4\end{smallmatrix}\right]$. Find a transition matrix consistent with these conditional expectations. Is this transition matrix unique (i.e. can you find another one that is consistent with these conditional expectations)?

We will use the content found in sectoin 2.2.6 (Enough one-step-ahead forecasts determine P). In our case we have that

$$h = \begin{bmatrix} 1 & 1 \\ 5 & 25 \end{bmatrix}$$

and that

$$J = \begin{bmatrix} 1.8 & 5.8 \\ 3.4 & 15.4 \end{bmatrix}$$

We can now use the expression $P = Jh^{-}1$ to solve for P (note that $h^{-}1 = \left[\begin{smallmatrix} 5.4 & -1/20 \\ -1/4 & 1/20 \end{smallmatrix}\right]$)

$$\begin{aligned} P &= Jh^{-1} \\ &= \begin{bmatrix} 1.8 & 5.8 \\ 3.4 & 15.4 \end{bmatrix} \begin{bmatrix} 5.4 & -1/20 \\ -1/4 & 1/20 \end{bmatrix} \\ &= \begin{bmatrix} .8 & .2 \\ .4 & .6 \end{bmatrix} \end{aligned}$$

□

Problem 2.3

Consumption is governed by an n-state Markov chain p, π_0 where P is a stochastic matrix and π_0 is an initial probability distribution. Consumption takes one of the values in the $nx1$ vector $\bar{c}$. A consumer ranks stochastic processes of consumption $t = 0, 1, \ldots$ according to

$$E \sum_{t=0}^{\infty} \beta^t u(c_t)$$

where E is the mathematical expectation and $u(c) = \frac{c^{1-\gamma}}{1-\gamma}$ for some parameter $\gamma \geq 1$. Let $u_i = u_i(\bar{c}_i)$. Let $v_i = E\left[\sum_{t=0}^{\infty} \beta^t u(c_t) | x_0 = \bar{e}_i\right]$ and $V = Ev$, where $\beta \in (0,1)$ is a discount factor.

a. Let u and v be the $nx1$ vectors whose ith components are u_i and v_i, respectively. Verify the following formulas for v and V: $v = (I - \beta P)^{-1} u$, and $V = \sum_i \pi_{0,i} v_i$.

b. Consider the following two markov processes
Process 1: $\pi_0 = \left[\begin{smallmatrix} .5 \\ .5 \end{smallmatrix}\right]$, $P = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$
Process 2: $\pi_0 = \left[\begin{smallmatrix} .5 \\ .5 \end{smallmatrix}\right]$, $P = \left[\begin{smallmatrix} .5 & .5 \\ .5 & .5 \end{smallmatrix}\right]$
For both Markov processes, $\bar{c} = \left[\begin{smallmatrix} 1 \\ 5 \end{smallmatrix}\right]$.
Assume that $\gamma = 2.5$ and $\beta = 0.95$. Compute the unconditional discounted expected utility V for each of these processes. Which of the two processes does the consumer prefer? Redo the calculations for $\gamma = 4$. Now which process does the consumer prefer?

c. An econometrician observes a sample of 10 observations of consumption rates for our consumer. He knows that one of the two preceding Markov processes generates the data, but he does not know which one. He assigns equal "prior probability" to the two chains. Suppose that the 10 successive observations on consumption are as follows: $1, 1, 1, 1, 1, 1, 1, 1, 1, 1$. Compute the likelihood of this sample under process 1 and under process 2. Denote the likelihood function Prob(Data|Model$_i$), $i = 1, 2$.

d. Suppose that the econometrician uses Bayes' law to revise his initial probability estimates for the two models,

Problem 2.3 continued on next page...

where in this context Bayes' law states:

$$\text{Prob}(M_i|data) = \frac{(\text{Prob(data)}|M_i)\cdot \text{Prob}(M_i)}{\sum_j \text{Prob(data)}|M_j \cdot \text{Prob}(M_j)}$$

where M_i denotes model i. The denominator of this expression is the unconditional probability of the data. After observing the data sample, what probabilities does the econometrician place on the two possible models?

e. Repeat the calculation in part d, but now assume that the data sample is $1,5,5,1,5,5,1,5,1,5$.

To start this solution, I wish to explicitly write out a few things:

- $c_t = \bar{c}' x_t$
- $v_i = E\left[\sum_{t=0}^{\infty} \beta^t u(c_t)|x_0 = \bar{e}_i\right] \rightarrow v_i = E\left[\sum_{t=0}^{\infty} \beta^t u(c_t)|c_0 = \bar{c}_i\right]$
- $u = u(\bar{c})$

a. For this part, I need to write v in vector notation using its component formulas

$$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} E\left[\sum_{t=0}^{\infty} \beta^t u(c_t)|c_0 = \bar{c}_1\right] \\ E\left[\sum_{t=0}^{\infty} \beta^t u(c_t)|c_0 = \bar{c}_2\right] \\ \vdots \\ E\left[\sum_{t=0}^{\infty} \beta^t u(c_t)|c_0 = \bar{c}_n\right] \end{bmatrix}$$

I will then apply the forecasting function given in section 2.2.5 of the notes (page 35), which reads (note that this function applies because $\beta \in (0,1)$):

$$\sum_{k=0}^{\infty} \beta^k E\left[y_{t+k}|x_t = \bar{e}_i\right] = \left[(I - \beta P)^{-1}\bar{y}\right]$$

In this case $y \rightarrow u(c)$, $\bar{y} \rightarrow u(\bar{c})$, and $x \rightarrow c$. Making these substitutions we obtain the desired equation:

$$v = (I - \beta P)^{-1} u$$

The other part of this problem is quite easy. We know that V is defined as Ev. To find this expected value we simply need to multiply on the left by the initial probability distribution π_0'. In mathematical terms we have:

$$V = Ev = \pi_0' v = \sum_i \pi_{0,i} v_i$$

Which is the desired result. □

b. To do this part I simply plugged things into the expressions derived in the previous part of the problem. I did this in the python program found below this problem and got that for both processes 1 and

2 the answers were $V = \begin{cases} -7.26295 & \text{when } \beta = 2.5 \\ -3.36 & \text{when } \beta = 4.0 \end{cases}$ Note that under both parameterizations of γ the consumer is indifferent between the two consumption processes. □

c. This part is simple. I simple need to use the expression for the likelihood function given in equation 2.2.15:

$$L = \text{Prob(data}|\text{ model}_i) = \pi_{0,i_0} \prod_i \prod_j P_{i,j}^{n_i j}$$

In my case this means $\text{Prob(data}|\text{ model}_i) = \left[\pi_{0,1}\right]_i \left[P_{1,1}^9\right]_i$, where $[\cdot]_i$ means for model i.

- Applying to process 1: $\text{Prob(data}|\text{ model}_1) = \left[\pi_{0,1}\right]_1 \left[P_{1,1}^9\right]_1 = (0.5)(1)^9 = 0.5$
- Applying to process 2: $\text{Prob(data}|\text{ model}_2) = \left[\pi_{0,1}\right]_2 \left[P_{1,1}^9\right]_2 = (0.5)(0.5)^9 = 9.766e-04$ □

d. This part is simply applying the given expression for Bayes' rule. Because we don't actually know the probability of either model, the solution will be in terms of those probabilities ($\text{Prob}[M_i]$). The denominator is the same for both models and is equal to:

$$\sum_j \text{Prob(data)}|M_j \cdot \text{Prob}(M_j) = 0.5 \cdot \text{Prob}(M_1) + 9.766e-04 \cdot \text{Prob}(M_2)$$

The numerators are different for each model, so I will show the final solution for each model separately:

- Model 1:

$$\begin{aligned} \text{Prob}(M_1|data) &= \frac{(\text{Prob(data)}|M_1) \cdot \text{Prob}(M_1)}{\sum_j \text{Prob(data)}|M_j \cdot \text{Prob}(M_j)} \\ &= \frac{0.5 \cdot \text{Prob}(M_1)}{0.5 \cdot \text{Prob}(M_1) + 9.766e-04 \cdot \text{Prob}(M_2)} \end{aligned}$$

- Model 2:

$$\begin{aligned} \text{Prob}(M_2|data) &= \frac{(\text{Prob(data)}|M_2) \cdot \text{Prob}(M_2)}{\sum_j \text{Prob(data)}|M_j \cdot \text{Prob}(M_j)} \\ &= \frac{9.766e-04 \cdot \text{Prob}(M_2)}{0.5 \cdot \text{Prob}(M_1) + 9.766e-04 \cdot \text{Prob}(M_2)} \end{aligned}$$

□

e. To do this part I need to apply the likelihood function to this new data. For each model I will need to evaluate the following expression (note I just use P here, but I will replace what with the actual P for each model):

$$L = (0.5)\,(P_{12})^4\,(P_{21})^3\,(P_{22})^2$$

- Model 1: $L = (0.5)(0)^4(0)^3(1)^2 = 0$
- Model 2: $L = (0.5)(0.5)^4(0.5)^3(0.5)^2 = 9.766e-04$

Now I apply Bayes' law:

- Model 1:

$$\begin{aligned}\text{Prob}(M_1|data) &= \frac{(\text{Prob}(\text{data})|M_1)\cdot\text{Prob}(M_1)}{\sum_j \text{Prob}(\text{data})|M_j\cdot\text{Prob}(M_j)} \\ &= \frac{0\cdot\text{Prob}(M_1)}{0\cdot\text{Prob}(M_1)+9.766e-04\cdot\text{Prob}(M_2)} \\ &= 0\end{aligned}$$

- Model 2:

$$\begin{aligned}\text{Prob}(M_2|data) &= \frac{(\text{Prob}(\text{data})|M_2)\cdot\text{Prob}(M_2)}{\sum_j \text{Prob}(\text{data})|M_j\cdot\text{Prob}(M_j)} \\ &= \frac{9.766e-04\cdot\text{Prob}(M_2)}{0\cdot\text{Prob}(M_1)+9.766e-04\cdot\text{Prob}(M_2)} \\ &= \frac{9.766e-04\cdot\text{Prob}(M_2)}{9.766e-04\cdot\text{Prob}(M_2)} \\ &= 1\end{aligned}$$

□

```
from __future__ import division
import numpy as np
from scipy.linalg import inv, eig
import pandas as pd
from rmt_utils import doublej
from matrix2latex import matrix2latex as to_tex


class StochasticLinearDiff(object):
    """
    Represents and computes various things for a model in the form
    of the canonical stochastic linear difference equation:

    .. math::

        x{t+1} = A x_t + C w_{t+1}
    """

    def __init__(self, A, C):
        self.A = A
        self.C = C

        # Evaluate eigenvalues and vectors for use later on. Check boundedness
        evals, evecs = eig(self.A, left=False, right=True)
        self.evals, self.evecs = evals, evecs
        self.unbounded = np.abs(evals).max() > 1

    @property
    def Cx(self):
        "Covariance stationary covariance matrix"
        if not self.unbounded:
            return doublej(self.A, self.C.dot(self.C.T))
        else:
            msg = 'This computation will not work because the eigenvalues'
            msg += '\nof A are not all below 1 in modulus.'
            raise ValueError(msg)

    @property
    def mu(self):
        "Covariance stationary mean"
        if self.unbounded:
            msg = 'This computation will not work because the eigenvalues {0}'
            msg += '\nof A are not all below 1 in modulus.'
            raise ValueError(msg.format(self.evals))

        # Try to get index of unit eigenvalue
        try:
```

```
            ind = np.where(self.evals == 1)[0][0]
        except IndexError:
            raise ValueError("The A matrix doesn't have any unit eigenvalues")

        # compute Stationary mean using the eigenvector for unit eigenvalue
        return self.evecs[:, ind] / self.evecs[-1, ind]


def isoelastic_util(c, gamma):
    """
    Implementation of the isoelastic utility function
    :math:`u = \\frac{c^{1 - \gamma}}{1 - \gamma}`
    """
    return c ** (1 - gamma) / (1 - gamma)


def p2_3b(P, beta=0.95, pi_0=[.5, .5], cbar=[1, 5], gamma=[2.5, 4.],
          u=isoelastic_util):
    """
    Solution to problem 2.3b for RMT4

    Parameters
    ==========
    P : iterable of array_like, dtype=float
```

Problem 2.4

Consider the univariate stochastic process

$$y_{t+1} = \alpha + \sum_{j=1}^{4} \rho_j y_{t+1-j} + c w_{t+1} \tag{1}$$

where w_{t+1} is a scalar martingale difference sequence adapted to $J_t = \left[w_t, \ldots w_1, y_0, y_{-1}, y_{-2}, y_{-3}\right]$, $\alpha = \mu\left(a - \sum_j \rho_j\right)$ and the ρ_j's are such that the matrix

$$A = \begin{bmatrix} \rho_1 & \rho_2 & \rho_3 & \rho_4 & \alpha \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

has all of its eigenvalues in modulus bounded below unity.

a. Show how to map this process into a first-order linear stochastic difference equation.

b. For each of the following examples, if possible, assume that the initial conditions are such that y_t is covariance stationary. For each case, state the appropriate initial conditions. Then compute the covariance stationary mean and variance of y_t assuming the following parameter sets of parameter values

   i. $\rho = [1.2, -0.3, 0, 0], \mu = 10, c = 1$

   ii. $\rho = [1.2, -0.3, 0, 0], \mu = 10, c = 2$

   iii. $\rho = [0.9, 0, 0, 0], \mu = 5, c = 1$

   iv. $\rho = [0.2, 0, 0, 0.5], \mu = 5, c = 1$

   v. $\rho = [0.8, 0.3, 0, 0], \mu = 5, c = 1$

   Hint 1: The Matlab program `doublej.m`, in particular, the command `X = doublej(A, C*C')` computes the solution to the matrix equation $AXA' + CC' = X$. This program is on Tom's website.

   Hint 2: The mean vector is the eigenvector of A associated with a unit eigen- value, scaled so that the mean of unity in the state vector is unity.

Problem 2.4 continued on next page...

c. For each case in part b, compute the h_j's in $E_t y_{t+5} = \gamma_0 + \sum_{j=0}^{3} h_j Y_{t-j}$.

d. For each case in part b, compute the $\tilde{h}_j$'s in $E_t \sum_{k=0}^{\infty} 0.95^k y_{t+k} = \sum_{j=0}^{3} \tilde{h}_j y_{t-j}$.

e. For each case in part b, compute the autocovariance $E(y_t - \mu_y)(y_{t-k} - \mu_y)$ for the three values $k = 1, 5, 10$.

a. Much of this part was already done for me when the matrix A was given above. I seek an equation in the form of 2.4.1. I show that below:

$$\begin{bmatrix} y_{t+1} \\ y_t \\ y_{t-1} \\ y_{t-2} \\ 1 \end{bmatrix} = \begin{bmatrix} \rho_1 & \rho_2 & \rho_3 & \rho_4 & \alpha \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_t \\ y_{t-1} \\ y_{t-2} \\ y_{t-3} \\ 1 \end{bmatrix} + \begin{bmatrix} c \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} w_{t+1}$$

$$y_{t+1} = A y_t + c w_{t+1}$$

□

b. To proceed on this part, I need to derive an expression for α in terms of ρ, μ, c. I start with 1 and take the unconditional expectation to obtain:

$$\mu = \alpha + \mu \sum_{j=1}^{4} \rho_j$$

which leads to

$$\alpha = \mu - \mu \sum_{j=1}^{4} \rho_j$$

In order for a stochastic process to be covariance stationary, two conditions must hold

(a) All eigenvalues of A (except perhaps 1 that corresponds to the constant term and is equal to unity) must be bounded below unity in modulus.

(b) The initial condition x_0 comes from a distribution described by the covariance stationary mean and covariance.

Now I will proceed with the calculations for each part. Note that I used a python adaptation of the code in `doublej.m` to compute the solutions as well as some new python code for this problem. They will both be included at the end of this problem.

i. The eigenvalues for A are $\lambda = [0., 0., 0.355, 0.845, 1.]$, which are all less than 1 in modulus (except the constant term), so there does exist a set of covariance stationary initial conditions. I found that they were equal to:

Problem 2.4 continued on next page...

$$\mu = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \\ 1 \end{bmatrix} \qquad C_x(0) = \begin{bmatrix} 7.42857 & 6.85714 & 6 & 5.14286 & 0 \\ 6.85714 & 7.42857 & 6.85714 & 6 & 0 \\ 6 & 6.85714 & 7.42857 & 6.85714 & 0 \\ 5.14286 & 6 & 6.85714 & 7.42857 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

ii. The eigenvalues for A are $\lambda = [0., 0., 0.355, 0.845, 1.]$, which are all less than 1 in modulus (except the constant term), so there does exist a set of covariance stationary initial conditions. I found that they were equal to:

$$\mu = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \\ 1 \end{bmatrix} \qquad C_x(0) = \begin{bmatrix} 29.7143 & 27.4286 & 24 & 20.5714 & 0 \\ 27.4286 & 29.7143 & 27.4286 & 24 & 0 \\ 24 & 27.4286 & 29.7143 & 27.4286 & 0 \\ 20.5714 & 24 & 27.4286 & 29.7143 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

iii. The eigenvalues for A are $\lambda = [0., 0., 0., 0.9, 1.]$, which are all less than 1 in modulus (except the constant term), so there does exist a set of covariance stationary initial conditions. I found that they were equal to:

$$\mu = \begin{bmatrix} 5 \\ 5 \\ 5 \\ 5 \\ 1 \end{bmatrix} \qquad C_x(0) = \begin{bmatrix} 5.26316 & 4.73684 & 4.26316 & 3.83684 & 0 \\ 4.73684 & 5.26316 & 4.73684 & 4.26316 & 0 \\ 4.26316 & 4.73684 & 5.26316 & 4.73684 & 0 \\ 3.83684 & 4.26316 & 4.73684 & 5.26316 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

iv. The eigenvalues for A are $\lambda = [0.895, 0.049+0.836i, 0.049-0.836i, -0.795, 1.]$, which are all less than 1 in modulus (except the constant term), so there does exist a set of covariance stationary initial conditions. I found that they were equal to:

$$\mu = \begin{bmatrix} 5 \\ 5 \\ 5 \\ 5 \\ 1 \end{bmatrix} \qquad C_x(0) = \begin{bmatrix} 1.4764 & 0.415887 & 0.166355 & 0.241214 & 0 \\ 0.415887 & 1.4764 & 0.415887 & 0.166355 & 0 \\ 0.166355 & 0.415887 & 1.4764 & 0.415887 & 0 \\ 0.241214 & 0.166355 & 0.415887 & 1.4764 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

v. The eigenvalues for A are $\lambda = [0., 0., -0.278233, 1.08, 1.]$, which are all not less than 1 in modulus (except the constant term), so there does not exist a set of covariance stationary initial conditions.

□

c. To solve this part of the problem, I set up a matrix equation to represent the expression given in the problem description. The equation I will work with is

$$EY = E[X\beta]$$

, where $Y = y_{t+5}$, $X = [y_t y_{t-1} y_{t-2} y_{t-3} 1]$ and β is a set of coefficients that represents γ_0 and each of the h_j's.

I can now obtain the standard linear least squares expression for β:

$$\beta = E\left[(XX')^{-1}\right] E[XY]$$

To actually get an expression for β I need to evaluate the two expectations in the expression above. I do this one at a time.

(a) $E\left[(XX')^{-1}\right]$: To evaluate this expression, I first turn to the definition of $C_x(0)$ given just below equation 2.4.10 on page 47:

$$\begin{aligned} C_x(0) &= E\left[\left(x_t - \mu\right)\left(x_t - \mu\right)'\right] \\ &= E[x_t x_t'] - \mu E[x_t]' - E[x_t]\mu' + \mu\mu' \\ &= E[x_t x_t'] - \mu\mu' \end{aligned}$$

Now I point out that in this case $x_t = X$, my matrix in the equation above. With this in mind it is clear that the expectation is:

$$E\left[(XX')^{-1}\right] = \left(C_x(0) + \mu\mu'\right)^{-1}$$

(b) $E[XY]$: For this part I set up a relationship between X and Y in the flavor equation 2.4.3b: $Y_t = X_t' G'$. Using this definition and the result from above, I can evaluate the expectation (note that I need $Y_{t+5} = X_{t+5}' G'$):

$$\begin{aligned} E[XY] &= E[X_t X_{t+5}' G'] \\ &= E[X_t X_{t-5}'] G' \\ &= \left(C_x(-5) + \mu\mu'\right) G' \\ &= \left(C_x(5)' + \mu\mu'\right) G' \\ &= \left(\left(A^5 C_x(0)\right)' + \mu\mu'\right) G' \\ &= \left(C_x(0)' A^{5'} + \mu\mu'\right) G' \end{aligned}$$

To evaluate the last few steps I used equation 2.4.11 as well as footnote 7 in RMT4.

Now, because I am working with $Y = y_{t+5}$, I identify $G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}$. Putting the two expressions for the expectations together I obtain a final expression for β that I can evaluate:

$$\beta = \left(C_x(0) + \mu\mu'\right)^{-1} \left(C_x(0)' A^{5'} + \mu\mu'\right) G'$$

I will apply this expression to each parameterization from part b:

i. $\beta = \begin{bmatrix} 0.739 & -0.260 & -0.000 & 0.000 & 5.216 \end{bmatrix}$

ii. $\beta = \begin{bmatrix} 0.739 & -0.260 & -0.000 & 0.000 & 5.216 \end{bmatrix}$

iii. $\beta = \begin{bmatrix} 0.590 & -0.000 & -0.000 & 0.000 & 2.048 \end{bmatrix}$

iv. $\beta = \begin{bmatrix} 0.200 & 0.020 & 0.004 & 0.251 & 2.624 \end{bmatrix}$

v. Not all eigenvalues are less than 1 in modulus, so there is no set of covariance stationary initial conditions, so there is no answer to this problem.

□

d. This problem is very similar to the previous one. The big difference is that on the LHS I have a sum, rather than just y_{t+5}. To handle this I use equation 2.4.18 and write the following:

$$E_t \sum_{j=0}^{\infty} 0.95^j y_{t+j} = G(I - 0.95A_0)^{-1} x_t$$

Now I am ready to let $Y = G(I - 0.95y_0)^{-1} x_t$ and move forward as before. Recall that I had this expression:

$$\beta = E\left[(XX')^{-1}\right] E[XY]$$

The first term ($E\left[(XX')^{-1}\right]$) is exactly the same as before, but to obtain the second term I need to insert the new expression for Y (note that when I do this I take the transpose as before):

$$\begin{aligned} E[XY] &= E\left[XX'(I - 0.95A)^{-1'} G'\right] \\ &- = E[XX](I - 0.95A)^{-1'} G' \\ &= \left(C_x(0) + \mu\mu'\right)(I - 0.95A)^{-1'} G' \end{aligned}$$

Putting the two expectations together, I derive the final expression for β:

$$\begin{aligned} \beta &= E\left[(XX')^{-1}\right] E[XY] \\ &= \left(C_x(0) + \mu\mu'\right)^{-1} \left(C_x(0) + \mu\mu'\right)(I - 0.95A)^{-1'} G' \\ &= (I - 0.95A)^{-1'} G' \end{aligned}$$

I then just plugged this in to the included python code and let it go to work on each parameter set to obtain beta.

(a) $\beta = \begin{bmatrix} 7.648 & -2.180 & 0.000 & 0.000 & 145.315 \end{bmatrix}$

(b) $\beta = \begin{bmatrix} 7.648 & -2.180 & 0.000 & 0.000 & 145.315 \end{bmatrix}$

(c) $\beta = \begin{bmatrix} 6.897 & -0.000 & -0.000 & 0.000 & 65.517 \end{bmatrix}$

(d) $\beta = \begin{bmatrix} 2.483 & 1.064 & 1.120 & 1.179 & 70.764 \end{bmatrix}$

(e) Not all eigenvalues are less than 1 in modulus, so there is no set of covariance stationary initial conditions, so there is no answer to this problem.

□

Problem 2.4 continued on next page...

e. The expectation in part e is $E(y_t - \mu_y)(y_{t-k} - \mu_y)$. This is simply an element of the autocovariance matrix $C_x(k)$. In fact, it is the (1, 1) element because $C_X(j)$ represents, in this problem, the autocovariance between the vectors y_t and y_{t-j}. To evaluate that term I simply used equation 2.4.11, which states:

$$C_x(j) = A_0^j C_x(0)$$

Doing the calculation for each parameterization and varying values of k I obtain the results in Table 1. □

```
from __future__ import division
import numpy as np
from scipy.linalg import inv, eig
import pandas as pd
from rmt_utils import doublej
from matrix2latex import matrix2latex as to_tex
        A list of transition matrices to be used as different processes
        consumption could follow.

    beta : float, optional(default=0.95)
        The value of the discount factor :math:`\\beta`

    pi_0: array_like, dtype=float, optional(default=[.5, 5])
        The value of the initial distribution :math:`\\pi_0`

    cbar : array_like, dtype=float, optional(default=[.5, 5])
        An array representing the consumption state space
        :math:`\\bar{c}`

    gamma : array_like, dtype=float, optional(default=[2.5, 4])
        Different values for the coeeficient of relative risk aversion
        :math:`gamma` in the utility function.

    u : function, optional(default=isoelastic_util)
        A python function representing the utility function. It is
        assumed that this function takes two arguments: 1. the value of
        consumption and 2. the value of gamma

    Returns
    =======
    V : pandas.DataFrame
        A pandas DataFrame representing the discounted expected utility
        :math:`V` given the parameters.
    """
    ## Define parameters
    beta = 0.95
    pi_0 = np.asarray(pi_0)
    cbar = np.asarray(cbar)

    # Prepare DataFrame to hold results
    c_names = ['Process%i' % (i + 1) for i in range(len(P))]
    V = pd.DataFrame(index=gamma, columns=c_names)

    for gam in gamma:
        u = cbar ** (1 - gam) / (1 - gam)
        for i, p in enumerate(P):
            v = inv(np.eye(2) - beta * p).dot(u)
            V.ix[gam, i] = v.dot(pi_0)

    return V

V = p2_3b(P=[np.eye(2), np.ones((2, 2)) * 0.5])
print(V.to_latex())

# Problem 2.4
case4_1 = ([1.2, -0.3, 0, 0], 10, 1)
case4_2 = ([1.2, -0.3, 0, 0], 10, 2)
case4_3 = ([0.9, 0, 0, 0], 5, 1)
case4_4 = ([0.2, 0, 0, 0.5], 5, 1)
case4_5 = ([0.8, 0.3, 0, 0], 5, 1)

cases4 = [case4_1, case4_2, case4_3, case4_4, case4_5]


def _beta2tex(beta):
    bet = pd.DataFrame(beta)
```

```
    return bet.T.applymap(lambda x: '%.3f' % x).to_latex(index=False,
                                                         header=False)


def p2_4(rho, mu, c, print_results=True):
    """
    Solution to problem 2.4 parts b-e for RMT4

    Problem Text
    ============

    Consider the univariate stochastic process

    .. math::

        y_{t+1} = \\alpha + \sum_{j=1}^4 \\rho_j y_{t+1 - j} + c w_{t+1}

    where :math:`w_{t+1}` is a scalar martingale difference sequence
    adapted to :math:`J_t = \left[ w_t, \dots w_1, y_0, y_{-1}, y_{-2},
    y_{-3} \\right], \\alpha = \mu \left( a - \sum_j \\rho_j \\right)` and
    the :math:`\\rho_j` 's are such that the matrix

    ..math ::

        \\begin{bmatrix}
        \\rho_1 & \\rho_2 & \\rho_3 & \\rho_4 & \\alpha \\
        1 &0 & 0 & 0 & 0 \\
        0 & 1 & 0 & 0 & 0 \\
        0 & 0 & 1 & 0 & 0 \\
        0 & 0 & 0 & 0 & 1
        \end{bmatrix}

    has all of its eigenvalues in modulus bounded below unity.

    # TODO: This is incomplete...

    Parameters
    ==========
    rho : array_like, dtype=float
        The array-like object that corresponds to
        :math:`[\\rho_1 \\rho_2 \\rho_3 \\rho_4]` in the problem

    mu : scalar, dtype=float
        The sclar :math:`\mu` from the problem

    c : scalar, dtype=float
        The value of the :math:`c` in the problem.

    print_results : bool
        Whether or not the function should print its results.

    Returns
    =======
    mu : array, dtype=float
        The array of average values. Represents the vector :math:`\mu`

    Cx : array, dtype=float
        The stationary variance/co-variance matrix for the problem
    """
    # Solve for alpha
    alpha = mu - mu * sum(rho)

    # Create a1 (A) matrix
    r1, r2, r3, r4 = rho

    a1 = np.array([[r1, r2, r3, r4, alpha],
                   [1, 0, 0, 0, 0],
                   [0, 1, 0, 0, 0],
                   [0, 0, 1, 0, 0],
                   [0, 0, 0, 0, 1],
                   ], dtype=float)

    # Check to make sure all eigenvalues are bounded below q
    w, vr = eig(a1, left=False, right=True)
    unbounded = np.abs(w).max() > 1
    if unbounded:
        msg = 'This computation will not work because the eigenvalues {0}\n'
        msg += 'of A are not all below 1 in modulus.'
        raise ValueError(msg.format(w))
```

Problem 2.4 continued on next page...

```
    # Try to get index of unit eigenvalue
    try:
        ind = np.where(w == 1)[0][0]
    except IndexError:
        raise ValueError('The A matrix does not have any unit eigenvalues')

    # compute Stationary mean using the eigenvector for unit eigenvalue
    mu = vr[:, ind] / vr[-1, ind]

    # Solve for the stationary covariance Cx
    # Create b1 (CC') matrix
    _c = np.array([c, 0, 0, 0, 0])
    b1 = np.outer(_c, _c)

    Cx = doublej(a1, b1)

    # Part c
    mumu = np.outer(mu, mu)
    G = np.array([1, 0, 0, 0, 0])
    term1 = inv(Cx + mumu)
    term2 = (np.dot(Cx.T, np.linalg.matrix_power(a1, 5).T) + mumu).dot(G)
    beta = np.dot(term1, term2)

    # Part d
```

Parameterization	i	ii	iii	iv	v
k					
1	6.857143	27.428571	4.736842	0.415887	NaN
5	3.702857	14.811429	3.107842	0.365232	NaN
10	1.597579	6.390317	1.835150	0.131579	NaN

Table 1: The autocovariances from problem 2.4 part e.

Problem 2.5

A consumer's rate of consumption follows the stochastic process

$$\begin{aligned} c_t + 1 &= \alpha_c + \sum_{j=1}^{2} \rho_j c_{t-j+1} + \sum_{j=1}^{2} \delta_j z_{t+1-j} + \psi_1 w_{1,t+1} \\ z_{t+1} &= \sum_{j=1}^{2} \gamma_j c_{t-j+1} + \sum_{j=1}^{2} \phi_j z_{t-j+1} + \psi_2 w_{2,t+1} \end{aligned} \tag{2}$$

wher W_{t+1} is a $2x1$ martingale difference sequence, adapted to $J_t = [w_t \dots w_1 c_0 c_{-1} z_0 z_{-1}]$, with contemporaneous covariance matrix $E w_{t+1} w'_{t+1} | J_t = I$, and the coefficients $\pi_j, \delta_j, \gamma_j, phi_j$ are such that the matrix

$$A = \begin{bmatrix} \rho_1 & \rho_2 & \delta_3 & \delta_4 & \alpha_c \\ 1 & 0 & 0 & 0 & 0 \\ \gamma_1 & \gamma_2 & \phi_1 & \phi_2 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

has all of its eigenvalues in modulus bounded below unity.

$$V_0 = E_0 \sum_{t=0}^{\infty} 0.95^t u - (c_t) \tag{3}$$

$$u(c_t) = -0.5(c_t - 60)^2 \tag{4}$$

a. Find a formula V_0 interms of the parameters of the one-period utility function (4) and the stochastic process for consumption.

b. Compute V_0 for the following two sets of parameter values:

   i. $\rho = [0.8 - 0.3]\,, \alpha_c = 1\delta = [0.20]\,, \gamma = [00]\,, \psi = [0.7 - 0.2]\,, \phi_1 = \phi_2 = 1$

   ii. The same as for part i except now $\phi_1 = 2, \phi_2 = 1$.

Hint: remember `doublej.m`.

a. To begin this part of the problem, I will first plug the expression for the one-period utility function into the equation for V_0 and expand/simplify:

$$\begin{aligned}
V_0 &= E_0 \sum_{t=0}^{\infty} 0.95^t u - (c_t) \\
&= E_0 \sum_{t=0}^{\infty} 0.95^t \left[-0.5(c_t - 60)^2\right] \\
&= E_0 \sum_{t=0}^{\infty} 0.95^t \left[-0.5(c_t^2 - 120c_t - 3600)\right] \\
&= E_0 \sum_{t=0}^{\infty} 0.95^t \left[-0.5c_t^2 + 60c_t - 1800)\right]
\end{aligned}$$

I now expand the summation from above by separating the $t = 0$ terms from all the $t > 0$ terms. Doing this allows me to write V_0 in the following recursive form:

$$\begin{aligned}
V_0 &= E_0 \sum_{t=0}^{\infty} 0.95^t \left[-0.5c_t^2 + 60c_t - 1800)\right] \\
&= E_0 0.95^0 \left[-0.5c_0^2 + 60c_0 - 1800\right] + E_0 \sum_{t=1}^{\infty} 0.95^t \left[-0.5c_t^2 + 60c_t - 1800)\right] \\
&= \left[-0.5c_0^2 + 60c_0 - 1800\right] + E_0 V_1
\end{aligned}$$

I can express the above expression as a matrix equation:

$$V_0 = x_0' G x_0 + 0.95 E_0 V_1 \tag{5}$$

where

Problem 2.5 continued on next page...

$$G = \begin{bmatrix} -0.5 & 0 & 0 & 0 & 30 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 30 & 0 & 0 & 0 & -1800 \end{bmatrix}$$

With this expansion, I see that the c_t^2 term and know that V_0 is quadratic in the state vector , so I will assume a general quadratic form for the expression of V_0:

$$V_0 = x'Rx + \xi$$

I will eventually need to evaluate the expectation $E_0 V_1$. Before doing this I need to define the stochastic process the form of the canonical stochastic difference equation . For this model, the objects in this equation are defined as follows:

$$\underbrace{\begin{bmatrix} c_{t+1} \\ c_t \\ z_{t+1} \\ z_t \\ 1 \end{bmatrix}}_{x_{t+1}} = \underbrace{\begin{bmatrix} \rho_1 & \rho_2 & \delta_1 & \delta_2 & \alpha_c \\ 1 & 0 & 0 & 0 & 0 \\ \gamma_1 & \gamma_2 & \phi_1 & \phi_2 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} c_t \\ c_{t-1} \\ z_t \\ z_{t-1} \\ 1 \end{bmatrix}}_{x_t} + \underbrace{\begin{bmatrix} \psi_1 & 0 \\ 0 & 0 \\ 0 & \psi_2 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} w_{1,t+1} \\ w_{2,t+1} \end{bmatrix}}_{C w_{t+1}}$$

I will now use the general quadratic form in addition to the matrices defined above to evaluate the expectation $E_0 V_1$ from Equation 5. Note two things:

1) I use the fact that from the definition of the martingale difference sequence, I know that $E[w_t] = 0$
2) I borrow from the matrix algebra expressed in section 2.4.5 of RMT4 explained in footnote 9 to move from an expression with w_1, R, C to an expression with the trace of R, C.

$$\begin{aligned} E_0 V_1 &= \xi + x_1' R x_1 \\ &= \xi + E_0 (Ax_0 + Cw_1)' R (Ax_0 + Cw_1) \\ &= \xi + E_0 x_0' A_0' R A x_0 + E_0 w_1' C' R A x_0 + E_0 x_0' A' R w_1 C + E_0 w_1' C' R C w_1 \\ &= \xi + x_0' A_0' R A x_0 + 0 + 0 + E_0 w_1' C' R C w_1 \\ &= \xi + x_0' A_0' R A x_0 + \text{trace}(RCC') \end{aligned}$$

I now substitute the expectation back into Equation 5 and see

$$V_0 = x_0' G x_0 + \xi + x_0' A_0' R A x_0 + \text{trace}(RCC')$$

I can now set this equal to the general quadratic form, equate terms, and uncover expressions for R and ξ.

$$V_0 = x'Rx + \xi = x_0' G x_0 + \beta \left[\xi + x_0' A_0' R A x_0 + \text{trace}(RCC')\right]$$
$$\vdots$$
$$R = G + \beta[A'RA]$$
$$d = \frac{\beta}{1-\beta}\text{trace}(C'RC)$$

Substituting these expressions back into the general quadratic form yields the desired expression for V_0. □

b. This part of the problem requires that I use the routine `doublej(sqrt(0.95) * A, G)` to obtain B. I can then plug this expression in to determine the value of d do this for each parameterization and show the results below:

(a)

$$R = \begin{bmatrix} -1.328\times10^{+05} & -1.280\times10^{+05} & 0.000\times10^{+00} & 0.000\times10^{+00} & -6.690\times10^{+04} \\ -1.280\times10^{+05} & -1.262\times10^{+05} & 0.000\times10^{+00} & 0.000\times10^{+00} & -6.356\times10^{+04} \\ 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} \\ 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} \\ -6.690\times10^{+04} & -6.356\times10^{+04} & 0.000\times10^{+00} & 0.000\times10^{+00} & -3.600\times10^{+04} \end{bmatrix}$$

$$\xi = -2.524 \times 10^{+06}$$

(b)

$$R = \begin{bmatrix} -1.328\times10^{+05} & -1.280\times10^{+05} & 0.000\times10^{+00} & 0.000\times10^{+00} & -6.690\times10^{+04} \\ -1.280\times10^{+05} & -1.262\times10^{+05} & 0.000\times10^{+00} & 0.000\times10^{+00} & -6.356\times10^{+04} \\ 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} \\ 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} & 0.000\times10^{+00} \\ -6.690\times10^{+04} & -6.356\times10^{+04} & 0.000\times10^{+00} & 0.000\times10^{+00} & -3.600\times10^{+04} \end{bmatrix}$$

$$\xi = -1.010e+07$$

□

```
from __future__ import division
import numpy as np
from scipy.linalg import inv, eig
import pandas as pd
from rmt_utils import doublej
from matrix2latex import matrix2latex as to_tex

    # Part e
    auto_covar = [np.linalg.matrix_power(a1, 1).dot(Cx)[0, 0],
                  np.linalg.matrix_power(a1, 5).dot(Cx)[0, 0],
                  np.linalg.matrix_power(a1, 10).dot(Cx)[0, 0]]

    if print_results:
        msg = 'Results for when rho = {0}, mu = {1} and c = {2}:'
        print(msg.format(rho, mu, c))
        evals = np.abs(w) if all(w == np.abs(w)) else w
        print('The eigenvalues are: %s' % evals)
        print('The stationary mean for this problem is:\n%s' % mu)
        print('The stationary covariance for this problem is:\n%s\n' % (Cx))
        print('The value of beta in part c is: \n%s\n' % (beta))
        print('The value of beta in part d is: \n%s\n' % (beta_d))
        print('The autocovariances are k=1: %.3f, k=5: %.3f, k=10: %.3f'
              % tuple(auto_covar))
```

```
    return mu, Cx, beta, beta_d, auto_covar

roman_numerals = ['i', 'ii', 'iii', 'iv', 'v']
auto_covars = pd.DataFrame(index=[1, 5, 10], columns=roman_numerals)
auto_covars.index.name = 'k'
auto_covars.columns.name = 'Parameterization'

for i, case in enumerate(cases4):
    try:
        print('------------ CASE {0} -------------'.format(i+1))
        # Get solution
        ux, Cx, beta_c, beta_d, a_cvar = p2_4(*case)
        auto_covars[roman_numerals[i]] = a_cvar

        # Print latex form for copy/paste
        msg = '\n\nmu:\n{0}\n\nCx:\n{1}\n\nbeta c:\n{2}\n\nbeta d:\n{3}'
        print(msg.format(to_tex(np.abs(ux)),
                         to_tex(Cx),
                         _beta2tex(beta_c),
                         _beta2tex(beta_d)))
    except ValueError as e: # don't let the eigenvalue error stop computation
        print('\n\n' + '*' * 72)
        err_msg = 'When rho = {0}, mu = {1} and c = {2} we had this error:'
        print(err_msg.format(*case))
        print(e)

print('\n\nThe autocovariances for part e are:\n ')
print(auto_covars.to_latex())


# Problem 2.5

# Define parameterizations. give (rho, alpha, delta, gamma, phi, psi1, psi2)
case5_1 = ([0.8, -0.3], 1., [0.2, 0], [0., 0.], [0.7, -0.2], 1., 1.)
case5_2 = ([0.8, -0.3], 1., [0.2, 0], [0., 0.], [0.7, -0.2], 2., 1.)

cases5 = [case5_1, case5_2]


def p2_5(rho, alpha, delta, gamma, phi, psi1, psi2):
    """
    Solution to problem 2.5 part b for RMT4

    Problem Text
    ============

    TODO: Fill this in

    Parameters
    ==========
    rho : array_like, dtype=float
        The value of rho from the problem description

    alpha : float
        The value of alpha from the problem description

    delta : array_like, dtype=float
        The value of delta from the problem description

    gamma : array_like, dtype=float
        The value of gamma from the problem description

    phi : array_like, dtype=float
        The value of phi from the problem description

    psi1 : float
        The value of psi1 from the problem description
```