352 Final Project

Kai Chang

1 One-Dimension Black and Scholes

1.1 Divergence Initial Form

Since $t = T - \tau$, $\tau = T - t$. Therefore,

$$\frac{\partial V}{\partial t} = \frac{\partial V}{\partial \tau}\frac{\partial \tau}{\partial t} = \frac{\partial V}{\partial \tau} \cdot -1 = -\frac{\partial V}{\partial \tau}$$

Multiplying the equation

$$\frac{\partial V}{\partial \tau} + \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 V}{\partial x^2} + rx\frac{\partial V}{\partial x} - rV = 0,$$

by -1 on both sides, we get

$$-\frac{\partial V}{\partial \tau} - \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 V}{\partial x^2} - rx\frac{\partial V}{\partial x} + rV = 0\,.$$

Substituting $-\dfrac{\partial V}{\partial \tau}$ by $\dfrac{\partial V}{\partial t}$, it yields

$$\frac{\partial V}{\partial t} - \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 V}{\partial x^2} - rx\frac{\partial V}{\partial x} + rV = 0\,. \tag{1}$$

Notice that

$$\begin{aligned}
&-\frac{\partial}{\partial x}\left(\mu(x)\frac{\partial V}{\partial x}\right) + \beta(x)\frac{\partial V}{\partial x} + \rho V \\
=& -\mu(x)\frac{\partial^2 V}{\partial x^2} - \frac{d\mu(x)}{dx}\frac{\partial V}{\partial x} + \beta(x)\frac{\partial V}{\partial x} + \rho V \\
=& -\mu(x)\frac{\partial^2 V}{\partial x^2} - \left(\frac{d\mu(x)}{dx} - \beta(x)\right)\frac{\partial V}{\partial x} + \rho V
\end{aligned}$$

Comparing the coefficients with those in Equation (1), we conclude that

$$\begin{gathered}
\mu(x) = \frac{1}{2}\sigma^2 x^2 \\
\frac{d\mu(x)}{dx} - \beta(x) = \sigma^2 x - \beta(x) = rx \\
\text{and } \rho = r\,.
\end{gathered}$$

Hence,

$$\begin{cases} \mu(x) = \frac{1}{2}\sigma^2 x^2 = \frac{1}{200}x^2 > 0 \\ \beta(x) = (\sigma^2 - r)x = -0.04x \\ \rho = r = 0.05 \end{cases}$$

and

$$\frac{\partial V}{\partial t} - \frac{\partial}{\partial x}\left(\frac{1}{200}x^2\frac{\partial V}{\partial x}\right) - 0.04x\frac{\partial V}{\partial x} + 0.05V = 0\,. \tag{2}$$

1.2 Weak Formulation[1]

The initial and boundary conditions read

$$\begin{aligned} &V(x,0) = \max(E - x, 0), \\ &V(0,t) = Ee^{-rt}, \\ &V(x_{\text{MAX}}, t) = 0 \end{aligned}$$

To get the weak formulation, we do the regular thing: multiply Equation (2) by a testing function w and integrate on the domain. This yields

$$\int_0^{100}\frac{\partial V}{\partial t}w dx - \frac{1}{200}\int_0^{100}\frac{\partial}{\partial x}\left(x^2\frac{\partial V}{\partial x}\right)w dx - 0.04\int_0^{100}x\frac{\partial V}{\partial x}w dx + 0.05\int_0^{100}Vw dx = 0\,.$$

We assume w is independent of time and vanishes on the boundary (not generally true). Then

$$\int_0^{100}\frac{\partial V}{\partial t}w dx = \frac{d}{dt}\int_0^{100}Vw dx$$

Integrating by parts leads to

$$-\frac{1}{200}\int_0^{100}\frac{\partial}{\partial x}\left(x^2\frac{\partial V}{\partial x}\right)w dx = \frac{1}{200}\int_0^{100}x^2\frac{\partial V}{\partial x}\frac{\partial w}{\partial x}dx\,.$$

Thus, the weak formulation reads

$$\frac{d}{dt}\int_0^{100}Vw dx + \frac{1}{200}\int_0^{100}x^2\frac{\partial V}{\partial x}\frac{dw}{dx}dx - 0.04\int_0^{100}x\frac{\partial V}{\partial x}w dx + 0.05\int_0^{100}Vw dx = 0 \tag{3}$$

[1]There is a version I did initially in Appendix I, in which I introduced a lifting function to deal with the non-homogeneous boundary condition but later found it is dealt properly if we assume the testing function vanishes on the boundary. The thing is I'm not sure whether the idea of using a lifting function makes sense and is necessary here. What also made me hesitated is: without a lifting function, the function V will not live in the same space as the testing function. Would the finite element method still make sense then? Because what I thought is we are using an approximate space to estimate V. I would really appreciate it if you may check the Appendix and leave a comment.

1.3 Finite Element Approximation with θ-Method

Assume we use linear finite elements. Let

$$V = \sum_{j=1}^{N} u_j(t)\phi_j(x)\,.$$

Substituting V and w in Equation (3), the finite element approximation reads

$$\frac{d}{dt}\left(\sum_{j=1}^{N} u_j(t)\int_0^{100}\phi_j\phi_i dx\right) + \sum_{j=1}^{N} u_j(t)(\frac{1}{200}\int_0^{100} x^2\frac{d\phi_j}{dx}\frac{d\phi_i}{dx}dx - 0.04\int_0^{100} x\frac{d\phi_j}{dx}\phi_i dx$$
$$+ 0.05\int_0^{100}\phi_j\phi_i dx) = 0\,.$$

This can be written in the form of

$$M\frac{du}{dt} = -Au$$

where

$$[M_{ij}] = \int_0^{100}\phi_j\phi_i dx$$

and

$$[A_{ij}] = \frac{1}{200}\int_0^{100} x^2\frac{d\phi_j}{dx}\frac{d\phi_i}{dx}dx - 0.04\int_0^{100} x\frac{d\phi_j}{dx}\phi_i dx + 0.05\int_0^{100}\phi_j\phi_i dx\,.$$

Recall that the $\theta-$method gives

$$M\frac{u^{n+1}-u^n}{\Delta t} = -(1-\theta)Au^n - \theta Au^{n+1}\,.$$

Therefore, the full discretization reads

$$(M + \Delta t\theta A)u^{n+1} = (M - \Delta t(1-\theta)A)u^n\,.$$

The convergence rate is expected to be $O(h + \Delta t^p)$ where $p = 2$ if $\theta = 0.5$ and $p = 1$ otherwise. (Since we are using linear finite elements here.)

1.4 Coding Choice

Based on manual computation,

$$M_{ij} = \begin{cases} h/6 & \text{if} \quad |i-j| = 1 \\ 4h/6 & \text{if} \quad i = j \\ 0 & \text{otherwise} \end{cases}$$

Divide A into three pieces: Af, Ad, Ar, where

$$[Af_{ij}] = \frac{1}{200}\int_0^{100} x^2\frac{d\phi_j}{dx}\frac{d\phi_i}{dx}dx\,,$$

$$[Ad_{ij}] = -0.04 \int_0^{100} x \frac{d\phi_j}{dx} \phi_i dx \,, \quad \text{and}$$

$$[Ar_{ij}] = 0.05 \int_0^{100} \phi_j \phi_i dx \,.$$

Again, based on manual computation,

$$Af_{ij} = \begin{cases} -h/600 * (3 * max\{i,j\}^2 + 3 * max\{i,j\} + 1) & \text{if} \quad |i-j| = 1 \\ h/600 * (6 * i^2 + 2) & \text{if} \quad i = j \\ 0 \quad \text{otherwise} \end{cases}$$

$$Ad_{ij} = \begin{cases} -0.04 * (i^2 + 3 * i + 1) * h/6 & \text{if} \quad i = j - 1 \\ -0.04 * (i^2 - 3 * i + 1) * h/6 & \text{if} \quad i = j + 1 \\ \quad \text{if} \quad i = j \\ 0 \quad \text{otherwise} \end{cases}$$

and

$$Ar = 0.05 \cdot M$$

In particular, we have

$$A = Af + Ad + Ar$$

The code I wrote is reported below.[2] So is the result of the code. I also tried the code fem1DTimeDependend-1.py. The result is also reported below.

```
def mass(N,h):
    return h * sp.diags([1/6, 4/6, 1/6], [-1,0,1], shape = [N,N], format =
    'csr')

def Af(xmax,h): # diffusion
    N = int(xmax/h)
    Af_ = np.zeros((N,N))
    for i in range(N):
        for j in range(N):
            if abs(i-j) > 1:
                Af_[i,j] = 0.
            elif i==j:
                Af_[i,j] = h/600*(6*i**2+2)
            else:
                idx = max(i,j)
                Af_[i,j] = -h/600*(3*idx**2+3*idx+1)
    return Af_

def Ad(xmax,h): # advection
    N = int(xmax/h)
    Ad_ = np.zeros((N,N))
    for i in range(N):
        for j in range(N):
            if abs(i-j)>1:
                Ad_[i,j] = 0
```

[2]The LaTeX format of how the code is presented here is borrowed from Diego Gonzalez. I am not sure if he used the same format for the final project; I just want to report this to avoid potential confusion.

```
            elif i==j:
                Ad_[i,j] = -0.04*(2*i**2+2)*h/6
            elif i<j:
                Ad_[i,j] = -0.04 * (i**2+3*i+1) * h/6
            else:
                Ad_[i,j] = -0.04 * (i**2-3*i+1)*h/6
    return Ad_

def Ar(xmax,h): # reaction (essentially same as mass)
    N = int(xmax)
    return 0.05 * mass(N,h)

def A(N,h):
    return Ad(N,h) + Af(N,h) + Ar(N,h)

def u0(xmax,h):
    N = int(xmax/h)
    u0_ = np.zeros((N))
    E = 30
    for i in range(N):
        u0_[i] = max(E-(i+1)*h,0)
    return u0_

def U(xmax, h, theta, dt, Tf):
    N = int(xmax/h)
    M = mass(N,h)
    A_ = A(N,h)
    u0_ = u0(xmax,h)
    n = int(Tf/dt)
    U = np.zeros((N,n+1))
    U[:,0] = u0_
    for i in range(n):
        leftMat = M+dt*theta*A_
        rightMat = M-dt*(1-theta)*A_
        un = rightMat*U[:,i].reshape(N,1)
        ut = solve(leftMat, un)
        U[:,i+1] = ut.reshape(N,)
    return U

## define problem
xmax = 100
h = 1
Tf = 1
dt = 0.01
theta = 0.5
U = U(xmax, h, theta, dt, Tf)
```

Listing 1: Code for 1-D FEM.

Compared with what we did in class, the shape is flipped, which should be expected since we applied a change of variable.

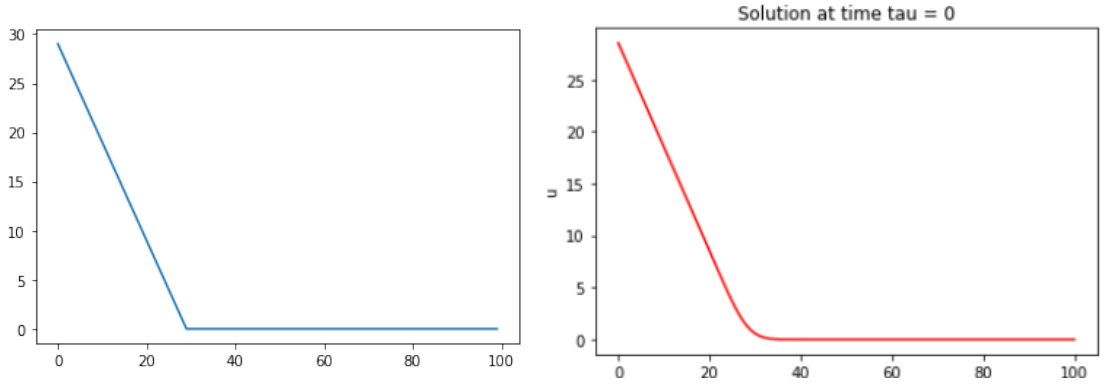

Figure 1: The result of my own code (left) and the result of the referred code (left).

2 Two-Dimension Black and Scholes

2.1 Finite Differences

Similar to the $1-D$ case, since $t = T - \tau$, $\tau = T - t$. Therefore,

$$\frac{\partial V}{\partial t} = \frac{\partial V}{\partial \tau}\frac{\partial \tau}{\partial t} = \frac{\partial V}{\partial \tau} \cdot -1 = -\frac{\partial V}{\partial \tau}$$

Therefore, replacing $\dfrac{\partial V}{\partial \tau}$ with $-\dfrac{\partial V}{\partial t}$,

$$\frac{\partial V}{\partial \tau} + \frac{1}{2}\sigma_1^2 x^2 \frac{\partial^2 V}{\partial x^2} + \frac{1}{2}\sigma_2^2 y^2 \frac{\partial^2 V}{\partial y^2} + \rho\sigma_1\sigma_2 xy \frac{\partial^2 V}{\partial x \partial y} + rx\frac{\partial V}{\partial x} + ry\frac{\partial V}{\partial y} - rV = 0$$

becomes

$$-\frac{\partial V}{\partial t} + \frac{1}{2}\sigma_1^2 x^2 \frac{\partial^2 V}{\partial x^2} + \frac{1}{2}\sigma_2^2 y^2 \frac{\partial^2 V}{\partial y^2} + \rho\sigma_1\sigma_2 xy \frac{\partial^2 V}{\partial x \partial y} + rx\frac{\partial V}{\partial x} + ry\frac{\partial V}{\partial y} - rV = 0\,.$$

Thus, the PDE with respect to $t = T - \tau$ reads

$$\frac{\partial V}{\partial t} - \frac{1}{2}\sigma_1^2 x^2 \frac{\partial^2 V}{\partial x^2} - \frac{1}{2}\sigma_2^2 y^2 \frac{\partial^2 V}{\partial y^2} - \rho\sigma_1\sigma_2 xy \frac{\partial^2 V}{\partial x \partial y} - rx\frac{\partial V}{\partial x} - ry\frac{\partial V}{\partial y} + rV = 0\,. \tag{4}$$

2.2 Bonus

By Taylor's Formula,

$$V(x+h, y+h) \approx V(x,y) + \frac{\partial V}{\partial x}(x,y)h + \frac{\partial V}{\partial y}(x,y)h + \frac{\partial^2 V}{\partial x^2}\frac{h^2}{2} + \frac{\partial^2 V}{\partial x \partial y}h^2 + \frac{\partial^2 V}{\partial y^2}\frac{h^2}{2} \tag{5}$$

$$+ \frac{\partial^3 V}{\partial x^3}\frac{h^3}{6} + \frac{\partial^3 V}{\partial x \partial y^2}\frac{h^3}{2} + \frac{\partial^3 V}{\partial x^2 \partial y}\frac{h^3}{2} + \frac{\partial^3 V}{\partial y^3}\frac{h^3}{6} + \frac{\partial^4 V}{\partial x \partial y^3}\frac{h^4}{6} + \ldots \tag{6}$$

$$V(x+h,y-h) \approx V(x,y) + \frac{\partial V}{\partial x}(x,y)h - \frac{\partial V}{\partial y}(x,y)h + \frac{\partial^2 V}{\partial x^2}\frac{h^2}{2} - \frac{\partial^2 V}{\partial x \partial y}h^2 + \frac{\partial^2 V}{\partial y^2}\frac{h^2}{2} \quad (7)$$

$$+ \frac{\partial^3 V}{\partial x^3}\frac{h^3}{6} + \frac{\partial^3 V}{\partial x \partial y^2}\frac{h^3}{2} - \frac{\partial^3 V}{\partial x^2 \partial y}\frac{h^3}{2} - \frac{\partial^3 V}{\partial y^3}\frac{h^3}{6} - \frac{\partial^4 V}{\partial x \partial y^3}\frac{h^4}{6} + ... \quad (8)$$

$$V(x-h,y+h) \approx V(x,y) - \frac{\partial V}{\partial x}(x,y)h + \frac{\partial V}{\partial y}(x,y)h + \frac{\partial^2 V}{\partial x^2}\frac{h^2}{2} - \frac{\partial^2 V}{\partial x \partial y}h^2 + \frac{\partial^2 V}{\partial y^2}\frac{h^2}{2} \quad (9)$$

$$- \frac{\partial^3 V}{\partial x^3}\frac{h^3}{6} - \frac{\partial^3 V}{\partial x \partial y^2}\frac{h^3}{2} + \frac{\partial^3 V}{\partial x^2 \partial y}\frac{h^3}{2} + \frac{\partial^3 V}{\partial y^3}\frac{h^3}{6} - \frac{\partial^4 V}{\partial x \partial y^3}\frac{h^4}{6} + ... \quad (10)$$

$$V(x-h,y-h) \approx V(x,y) - \frac{\partial V}{\partial x}(x,y)h - \frac{\partial V}{\partial y}(x,y)h + \frac{\partial^2 V}{\partial x^2}\frac{h^2}{2} + \frac{\partial^2 V}{\partial x \partial y}h^2 + \frac{\partial^2 V}{\partial y^2}\frac{h^2}{2} \quad (11)$$

$$- \frac{\partial^3 V}{\partial x^3}\frac{h^3}{6} - \frac{\partial^3 V}{\partial x \partial y^2}\frac{h^3}{2} - \frac{\partial^3 V}{\partial x^2 \partial y}\frac{h^3}{2} - \frac{\partial^3 V}{\partial y^3}\frac{h^3}{6} + \frac{\partial^4 V}{\partial x \partial y^3}\frac{h^4}{6} + ... \quad (12)$$

I did not write out other 4^{th} order terms because even for $\frac{\partial^4 V}{\partial x \partial y^3}$, it cannot be cancelled out. Thus, (5) - (7) - (9) + (11) reads

$$V(x+h,y+h) - V(x+h,y-h) - V(x-h,y+h) + V(x-h,y-h) \approx 4\frac{\partial^2 V}{\partial x \partial y}h^2 + O(h^4) \quad (13)$$

Equation (13) gives

$$\frac{\partial^2 V}{\partial x \partial y} \approx \frac{V(x+h,y+h) - V(x+h,y-h) - V(x-h,y+h) + V(x-h,y-h)}{4h^2} + O(h^2)\,.$$

Therefore, the formula gives quadratic accuracy.

2.3 Finite Differences Continued

Apply 2^{nd}-order Finite Differences on $\frac{\partial^2 V}{\partial x^2}$, $\frac{\partial^2 V}{\partial y^2}$, $\frac{\partial^2 V}{\partial x \partial y}$, $\frac{\partial V}{\partial x}$, and $\frac{\partial V}{\partial y}$ respectively. In particular,

$$\frac{\partial^2 V}{\partial x^2}(x_i,y_j) \approx \frac{V_{i+1,j} - 2V_{i,j} + V_{i-1,j}}{h^2}$$

$$\frac{\partial^2 V}{\partial y^2}(x_i,y_j) \approx \frac{V_{i,j+1} - 2V_{i,j} + V_{i,j-1}}{h^2}$$

$$\frac{\partial V}{\partial x}(x_i,y_j) \approx \frac{V_{i+1,j} - V_{i-1,j}}{2h}$$

$$\frac{\partial V}{\partial y}(x_i, y_j) \approx \frac{V_{i,j+1} - V_{i,j-1}}{2h}$$

and $\dfrac{\partial^2 V}{\partial x \partial y}$ admits the discretization formula just mentioned. Then the whole discretization on space should have second-order accuracy.

Coding Choice:

Since this is a 2-D problem, we want to store all $V_{i,j}$ (including boundary cases) in a 1-D vector of size $(N+1)^2$ where $N = xmax/h$. To write the part of the system that does not involve time in the form of AV where V is the vector that stores all the $V_{i,j}$, we need A to be of size $(N+1)^2 \times (N+1)^2$. The code that build the matrix A is presented below.

```
# import packages
import numpy as np
import scipy.sparse as sp
import pandas as pd
import matplotlib.pyplot as plt
from numpy.linalg import solve

# build matrix A
def buildA(spaceMax, h):
    sigma1 = 0.1
    sigma2 = 0.3
    rho = 0.5
    r = 0.1
    N = int(spaceMax/h)+1
    A = np.zeros((N*N,N*N)) # size (N-1) * (N-1)

    for j in range(N): # adjust boundary coefficients
        A[j,j] = 1
        A[-j,-j] = 1
        A[j*N, j*N] = 1
        A[(j+1)*(N-1), (j+1)*(N-1)] = 1

    for i in range(1,N-1):
        xi = i*h
        for j in range(1,N-1):
            yj = j*h
            #V_{i}{j}
            A[i*N+j, i*N+j] = sigma1**2*xi**2/h**2+sigma2**2*yj**2/h**2+r
            #V_{i}{j-1}
            A[i*N+j, i*N+j-1] = -0.5 * (sigma2 * yj / h)**2 + r*yj/(2*h)
            #V_{i}{j+1}
            A[i*N+j, i*N+j+1] = -0.5 * (sigma2 * yj / h)**2 - r*yj/(2*h)
            #V_{i+1}{j}
            A[i*N+j, (i+1)*N + j] = -0.5 * (sigma1*xi/h)**2 - r*xi/(2*h)
            #V_{i+1}{j-1}
            A[i*N+j, (i+1)*N + j-1] = rho*sigma1*sigma2*xi*yj/(4*h**2)
            #V_{i+1}{j+1}
            A[i*N+j, (i+1)*N + j+1] = -rho*sigma1*sigma2*xi*yj/(4*h**2)
            #V_{i-1}{j}
            A[i*N+j, (i-1)*N + j] = -0.5 * (sigma1*xi/h)**2 + r*xi/(2*h)
            #V_{i-1}{j-1}
            A[i*N+j, (i-1)*N + j-1] = -rho*sigma1*sigma2*xi*yj/(4*h**2)
```

```
            #V_{i-1}{j+1}
            A[i*N+j, (i-1)*N + j+1] = rho*sigma1*sigma2*xi*yj/(4*h**2)

    return A
```

Listing 2: Matrix Builder

Notice that due to the adjustments of boundary conditions, there is another vector b in the system whose value depends on time. Here we present the code for constructing the vector b.

```
def V_0y(j,t,h):
    return gy(j*h,t) # boundary condition

def V_x0(i,t,h):
    return gx(i*h,t) # boundary condition

def vec(t,spaceMax,h): # this is the vector generated naturally in the
    system due to adjustment of boundary contitions
    N = int(spaceMax/h)+1
    b = np.zeros((N*N,1))
    for j in range(N):
        b[j] = V_0y(j,t,h)
        b[-j] = 0. # for the sake of completeness
    for i in range(N):
        b[i*N] = V_x0(i,t,h)
        b[(i+1)*(N-1)] = 0 # for the sake of completeness
    return b
```

Listing 3: Vector Builder

We use $\theta-$method here for time discretization. First we write the system in the form of

$$\frac{\partial V}{\partial t} = -AV - b$$

The $\theta-$method reads

$$\frac{V^{n+1} - V^n}{\Delta t} = -(1-\theta)AV^n - (1-\theta)b^n - \theta AV^{n+1} - \theta b^{n+1}$$

which is equivalent to

$$(I + \Delta t\theta A)V^{n+1} = (I - \Delta t(1-\theta)A)V^n - (1-\theta)b^n - \theta b^{n+1}\,.$$

The code is presented below.

```
# IC
def IC(x,y):
    E=30
    return max(0, E-(x+y))

# the initial vector
def V_t0(spaceMax, h):
    N = int(spaceMax/h)+1
    v0 = np.zeros((N*N,1))
```

```
    for i in range(N):
        xi = i*h
        for j in range(N):
            yi = j*h
            v0[i*N+j] = IC(xi,yj)

    return v0

def fullDiscretization(Tf, dt, theta, spaceMax, h):
    N = int(spaceMax/h)+1 # number of space nodes
    n = int(Tf/dt) # number of time nodes
    A = buildA(spaceMax,h) # coefficient matrix
    v0 = V_t0(spaceMax,h) # initial condition
    V = v0
    I = np.identity(N**2) # identity matrix of size N^2
    leftMat = I + dt*theta*A
    rightMat = I - dt*(1-theta)*A
    for i in range(n): # loop over time
        ti = (i+1)*dt
        rightVec = vec(ti,spaceMax,h)
        vn = V[:,i] # retrieve vn
        RHS = np.dot(rightMax,vn) + rightVec # compute the right hand side
        vt = solve(leftMat, RHS) # solve for v_{n+1}
        V = np.append(V, vt, axis=1) # append the result

    return V
```

Listing 4: Full Discretization

One thing I hope to note is the boundary condition in my code is missing. That is why I am not able to report the result here. But the code should work if the boundary conditions are well-defined as expected. In short, if $gx(x,t)$ and $gy(y,t)$ is well-implemented, then everything should be fine.

2.4 Finite Elements

1) We want to show Equation (4) is equivalent to the "divergence" initial value form of

$$\frac{\partial V}{\partial t} - \nabla \cdot (\mathbb{T}\nabla V) + \boldsymbol{\beta} \cdot \nabla V + rV = 0 \tag{14}$$

where

$$\mathbb{T} \equiv \frac{1}{2} \begin{pmatrix} \sigma_1^2 x^2 & \rho\sigma_1\sigma_2 xy \\ \rho\sigma_1\sigma_2 xy & \sigma_2^2 y^2 \end{pmatrix}$$

is a matrix (or tensor) and

$$\boldsymbol{\beta} \equiv \begin{pmatrix} \left(\sigma_1^2 + \frac{1}{2}\rho\sigma_1\sigma_2 - r\right) x \\ \left(\sigma_2^2 + \frac{1}{2}\rho\sigma_1\sigma_2 - r\right) y \end{pmatrix}$$

is a vector.
Note that

$$\begin{aligned}\nabla \cdot (\mathbb{T}\nabla V) &= \frac{\partial}{\partial x}\left(\frac{1}{2}\left(\sigma_1^2 x^2 \frac{\partial V}{\partial x} + \rho\sigma_1\sigma_2 xy\frac{\partial V}{\partial y}\right)\right) + \frac{\partial}{\partial y}\left(\frac{1}{2}\left(\rho\sigma_1\sigma_2 xy\frac{\partial V}{\partial x} + \sigma_2^2 y^2\frac{\partial V}{\partial y}\right)\right)\\ &= \left(\sigma_1^2 x\frac{\partial V}{\partial x} + \frac{1}{2}\sigma_1^2 x^2\frac{\partial^2 V}{\partial x^2} + \frac{1}{2}\rho\sigma_1\sigma_2 y\frac{\partial V}{\partial y} + \frac{1}{2}\rho\sigma_1\sigma_2 xy\frac{\partial^2 V}{\partial x\partial y}\right)\\ &+ \left(\frac{1}{2}\rho\sigma_1\sigma_2 x\frac{\partial V}{\partial x} + \frac{1}{2}\rho\sigma_1\sigma_2 xy\frac{\partial^2 V}{\partial x\partial y} + \sigma_2^2 y\frac{\partial V}{\partial y} + \frac{1}{2}\sigma_2^2 x^2\frac{\partial^2 V}{\partial y^2}\right)\\ &= (\sigma_1^2 x + \frac{1}{2}\rho\sigma_1\sigma_2 x)\frac{\partial V}{\partial x} + (\sigma_2^2 y + \frac{1}{2}\rho\sigma_1\sigma_2 y)\frac{\partial V}{\partial y} + \rho\sigma_1\sigma_2 xy\frac{\partial^2 V}{\partial x\partial y} + \frac{1}{2}\sigma_1^2 x^2\frac{\partial^2 V}{\partial x^2} + \frac{1}{2}\sigma_2^2 x^2\frac{\partial^2 V}{\partial y^2}\end{aligned}$$

and that

$$\boldsymbol{\beta}\cdot\nabla V = \frac{\partial V}{\partial x}\left(\sigma_1^2 + \frac{1}{2}\rho\sigma_1\sigma_2 - r\right)x + \frac{\partial V}{\partial y}\left(\sigma_2^2 + \frac{1}{2}\rho\sigma_1\sigma_2 - r\right)y\,.$$

Therefore, Equation (14) becomes

$$\frac{\partial V}{\partial t} - \rho\sigma_1\sigma_2 xy\frac{\partial^2 V}{\partial x\partial y} - \frac{1}{2}\sigma_1^2 x^2\frac{\partial^2 V}{\partial x^2} - \frac{1}{2}\sigma_2^2 x^2\frac{\partial^2 V}{\partial y^2} - rx\frac{\partial V}{\partial x} - ry\frac{\partial V}{\partial y} + rV = 0$$

which is equivalent to Equation (4).

2) We do the usual set-up for weak formulation: multiply by a test function in the functional space and integrate the whole equation over the domain. That is,

$$\int_\Omega \frac{\partial V}{\partial t}\phi dw - \int_\Omega \nabla\cdot(\mathbb{T}\nabla V)\phi dw + \int_\Omega \boldsymbol{\beta}\cdot\nabla V\phi dw + r\int_\Omega V\phi dw = 0$$

where $dw = dxdy$ in this case. Since ϕ vanishes on the boundary and ϕ is independent of t, the formulation then reads

$$\frac{d}{dt}\int_\Omega V\phi dw + \int_\Omega \mathbb{T}\nabla V\cdot\nabla\phi dw + \int_\Omega \boldsymbol{\beta}\cdot\nabla V\phi dw + r\int_\Omega V\phi dw = 0$$

3) We proceed with Finite Element Method. First, discretize the space. Let

$$V = \sum_{j=1}^{N} u_j(t)\phi_j(x,y)\,.$$

The finite element approximation reads

$$\sum_{j=1}^{N}\frac{d}{dt}\left(u_j(t)\int_\Omega \phi_j\phi_i dw\right) + \sum_{j=1}^{N} u_j(t)\left(\int_\Omega \mathbb{T}\nabla\phi_j\cdot\nabla\phi_i dw + \int_\Omega \boldsymbol{\beta}\cdot\nabla\phi_j\phi_i dw + r\int_\Omega \phi_j\phi_i dw\right) = 0$$

which can be written in the form of

$$M\frac{du}{dt} + Au = 0$$

where

$$[M_{ij}] = \int_\Omega \phi_j \phi_i dw$$

and

$$[A_{ij}] = \int_\Omega \mathbb{T}\nabla\phi_j \cdot \nabla\phi_i dw + \int_\Omega \boldsymbol{\beta} \cdot \nabla\phi_j \phi_i dw + r\int_\Omega \phi_j \phi_i dw\,.$$

Now we apply $\theta-$method to discretize the time variable. Recall the $\theta-$method reads

$$M\frac{u^{n+1} - u^n}{\Delta t} = -(1-\theta)Au^n - \theta Au^{n+1}\,.$$

Therefore, the full discretization reads

$$(M + \Delta t\theta A)u^{n+1} = (M - \Delta t(1-\theta)A)u^n\,.$$

The convergence rate is expected to be $O(h^q + \Delta t^p)$ where $p = 2$ if $\theta = 0.5$ and $p = 1$ otherwise. Here q is the degree of our finite elements since we are assuming the solution lives in such a functional space that is infinitely regular.

Appendix I

The initial and boundary conditions read

$$\begin{aligned} &V(x,0) = \max(E-x,0), \\ &V(0,t) = Ee^{-rt}, \\ &V(x_{\text{MAX}},t) = 0 \end{aligned}$$

For simplicity of notation, let $\mathcal{L}^2(H^1)$ denote the functional space

$$\mathcal{L}^2(0,1;H^1(0,100)) = \{f \in H^1(0,100) \ | \ ||f||_{H^1}(t) \in \mathcal{L}^2(0,1)\}$$

where $\mathcal{L}$ and H are the notations for *Lebesgue Spaces* and *Sobolev Spaces* respectively. Define the functional space $\mathcal{V}$ as

$$\mathcal{V} = \{f \in \mathcal{L}^2(H^1) \ | \ f(0,t) = f(100,t) = 0\}\,.$$

Write V (the function of concern) as $V = \widetilde{V} + l$ where $l = l(x,t)$ satisfies $l(0,t) = Ee^{-rt}$ and $l(100,t) = 0$ and $\widetilde{V} \in \mathcal{V}$.
Replacing V with $\widetilde{V} + l$ in Equation 2 and simplifying the equation, it yields

$$\frac{\partial(\widetilde{V}+l)}{\partial t} - \frac{\partial}{\partial x}\left(\frac{1}{200}x^2\frac{\partial(\widetilde{V}+l)}{\partial x}\right) - 0.04x\frac{\partial(\widetilde{V}+l)}{\partial x} + 0.05(\widetilde{V}+l) = 0\,. \tag{15}$$

Multiply it by a function $w \in \mathcal{V}$ and integrate the resultant equation on both sides with respect to x from 0 to 100:

$$\int_0^{100}\frac{\partial(\widetilde{V}+l)}{\partial t}wdx - \frac{1}{200}\int_0^{100}\frac{\partial}{\partial x}\left(x^2\frac{\partial(\widetilde{V}+l)}{\partial x}\right)wdx - 0.04\int_0^{100}x\frac{\partial(\widetilde{V}+l)}{\partial x}wdx + 0.05\int_0^{100}(\widetilde{V}+l)wdx = 0\,.$$

We assume w is independent of time (not generally true). Then

$$\int_0^{100}\frac{\partial(\widetilde{V}+l)}{\partial t}wdx = \frac{d}{dt}\int_0^{100}(\widetilde{V}+l)wdx$$

Integrating by parts leads to

$$-\frac{1}{200}\int_0^{100}\frac{\partial}{\partial x}\left(x^2\frac{\partial(\widetilde{V}+l)}{\partial x}\right)wdx = \frac{1}{200}\int_0^{100}x^2\frac{\partial(\widetilde{V}+l)}{\partial x}\frac{\partial w}{\partial x}dx\,.$$

We let

$$(v,w) = \int_0^{100}vwdx$$

and

$$a(v,w) = \frac{1}{200}\int_0^{100}x^2\frac{\partial v}{\partial x}\frac{\partial w}{\partial x}dx - 0.04\int_0^{100}x\frac{\partial v}{\partial x}wdx + 0.05\int_0^{100}vwdx\,.$$

The problem then becomes as the following: find $\widetilde{V} \in \mathcal{V}$ such that for all $w \in \mathcal{V}$ such that

$$\frac{d}{dt}(\widetilde{V},w) + a(\widetilde{V},w) = -a(l,w)$$

where $l = l(x,t)$ is a selected function that satisfies $l(0,t) = Ee^{-rt}$ and $l(100,t) = 0$.