nmi | spring 2024

lecture 20 : nonlinear

8.4 nonlinear partial differential equations

the implicit BDM method of section 8.1 will be the working example for nonlinear FDM, FEM.

8.4.1 implicit newton solver

burgers equation is an elliptic equation named after jm burgers that is a simpliflied model of fluid flow, with visocity D. it is a typical nonlinear example and a benchmark for solvers.

discretize burgers as the heat equation was in section 8.1. denote approximate solution at (x_i, t_i) by w_{ij}. let M, N be number of total steps in x, t directions. apply BD to u_t and CD to other terms.

$$\frac{w_{ij} - w_{i,j-1}}{k} + w_{i,j}\frac{w_{i+1,j} - w_{i-1,j}}{2h} = \frac{D}{h^2}(w_{i+1,j} - 2w_{ij} + w_{i-1,j})$$

$$\Downarrow$$

$$w_{ij} + \frac{k}{2h}w_{ij}(w_{i+1,j} - w_{i-1,j}) - \sigma(w_{i+1,j} - 2w_{i,j} + w_{i-1,j}) - w_{i,j-1} = 0, \quad \sigma = \frac{Dk}{h^2}$$

bc of the quadratics, $w_{i+1,j}, w_{ij}, w_{i-1,j}$ cannot be solved explicitly or implicitly, so use multivariate newton (ch 2).

for one j at a time, given time j, define $z_i = w_{ij}$.

$$F_i(z_i, \dots, z_m) = z_i + \frac{k}{2h}z_i(z_{i+1} - z_{i-1}) - \sigma(z_{i+1} - 2z_9 + z_{i-1}) - w_{i,j-1} = 0$$

for m unknowns $z_1, \ldots, z_m$. oc $w_{i,j-1}$ is known. and first and last equations are replaced by boundary conditions. eg, for burgers

$$\begin{cases} {}_t + uu_x = Du_{xx} \\ u(x,0) = f(x), \quad x_l \le x \le x_r \\ u(x_l,t) = l(t), \quad t \ge 0 \\ u(x_r,t) = r(t), \quad t \ge 0 \\ \\ F_1(z_1,\ldots,z_m) = z_1 - l(t_j) = 0 \\ F_m(z_1,\ldots,z_m) = z_m - r(t_j) = 0 \end{cases}$$

ie, there are m nonlinear algebraic equations in m unknowns.

to apply multivariate newton, compute jacobian $DF(\vec{z}) = \frac{\partial \vec{F}}{\partial \vec{z}}$

$$\begin{bmatrix} 1 & 0 & \cdots & & \\ -\sigma - \frac{kz_2}{2h} & 1 + 2 + \frac{k(z_3 - z_1)}{2h} & -\sigma + \frac{kz_2}{2h} & & \\ & -\sigma - \frac{kz_3}{2h} & 1 + 2 + \frac{k(z_4 - z_2)}{2h} & -\sigma + \frac{kz_3}{2h} & \\ & & \ddots & \ddots & \ddots \\ & & & -\sigma - \frac{kz_{m-1}}{2h} & 1 + 2 + \frac{k(z_m - z_{m-2})}{2h} & -\sigma \\ & & & \cdots & 0 \end{bmatrix}$$

and solve for $z_i = w_{ij}$ by multivariate newton iteration

$$\vec{z}^{K+1} = \vec{z}^K - DF(\vec{z}^K)^{-1} F(\vec{z}^K).$$

▾ example 12

use BDE with MNI to solve burgers.

$$
\begin{cases}
_t + uu_x = Du_{xx} \\
u(x,0) = \frac{2D\beta\pi sin\pi x}{\alpha+\beta cos\pi x}, \quad 0 \le x \le 1, \alpha = 5, \beta = 4 \\
u(0,t) = 0, \quad t \ge 0 \\
u(1,t) = 0, \quad t \ge 0
\end{cases}
$$

> code, matlab

↳ *1 cell hidden*

> code, python

[ ] ↳ *3 cells hidden*

v usw

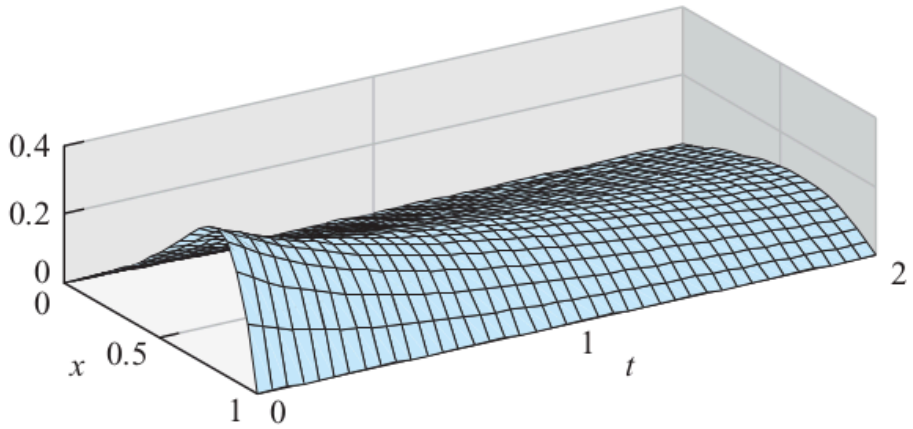

Figure 8.19 Approximate solution to Burgers' equation (8.66). Homogeneous Dirichlet boundary conditions are assumed, with step sizes $h = k = 0.05$.

v usw

consider polynomial $P(x_1, x_2, x_3) = x_1 x_2 x_3^2 + x_1^4$, which is called homogeneous of degree four since it consists entirely of degree four terms in x_1, x_2, x_3. the partial of P contained in gradient

$$\nabla P = (x_2 x_3^2 + 4x_1^3, x_1 x_3^2, 2x_1 x_2 x_3).$$

recover P by multiplying gradient by the vector of variables with an extra multiple of its order, 4.

$$\nabla P \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = (x_2 x_3^2 + 4x_1^3)x_1 + x_1 x_3^2 x_2 + 2x_1 x_2 x_3 x_3 = 4x_1 x_2 x_3^2 + 4x_1^4 = 4P.$$

in general, define polynomial $P(x_1, \ldots, x_m)$ to be **homogeneous** of degree d if

$$P(cx_1, \ldots, cx_m) = c^d P(x_1, \ldots, x_m), \quad \text{for all } c.$$

lemma 11

let $P(x_1, \ldots, x_m)$ be homogeneous polynomial of degree d. then

$$\nabla P \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = dP.$$

proof

differentiating $P(cx_1, \ldots, cx_m) = c^d P(x_1, \ldots, x_m)$ wrt c yields

$$x_1 P_{x_1}(cx_1, \ldots, cx_m) + \cdots + x_m P_{x_m}(cx_1, \ldots, cx_m) = dc^{d-1} P(x_1, \ldots, x_m).$$

evaluate at $c = 1$. ■

usw

ie, code for PDEs with polynomials can be written compactly if like degrees grouped together.

for certain boundary conditions, an explicit solution for burgers equation is known. the solution to the dirichlet problem is

$$u(x,t) = \frac{2D\beta\pi e^{-D\pi^2 t} sin\pi x}{\alpha + \beta e^{-D\pi^2 t} cos\pi x}.$$

ie, use that to measure accuracy of approximation per h, k. eg, $\alpha = 5, \beta = 4, D = 0.05$ at $x = \frac{1}{2}$ after one time step,

h	k	$u(0.5, 1)$	$w(0.5, 1)$	error
0.01	0.04	0.153435	0.154624	0.001189
0.01	0.02	0.153435	0.154044	0.000609
0.01	0.01	0.153435	0.153749	0.000314

which shows a first-order decrease in error as a function of time step size k, as expected with BDM.

consider **reaction-diffusion equations**. r.a. fisher is a successor of darwin who helped create the foundations of modern statistics. **fishers equation** was originally derived to model how genes propagate.

$$u_t = \underbrace{Du_{xx}}_{\text{diffusion}} + \underbrace{f(u)}_{\text{reaction}}$$

where $f(u)$ is a polynomial in u. if homogeneous neumann boundary conditions are used, the constant - ie, equilibrium - state $u(x,t) \equiv C$ is the solution whenever $f(C) = 0$. the equilibrium state is stable if $f(C) < 0$. ie, nearby solutions tend toward the equilibrium state.

<details>
<summary>example 13</summary>

use BDE with MNI to solve fishers equation with homogeneous neumann boundary conditions

$$\begin{cases} u_t = Du_{xx} + u(1-u) \\ u(x,0) = sin\pi x, \quad 0 \leq x \leq 1 \\ u_x(0,t) = 0, \quad t \geq 0 \\ u_x(1,t) = 0, \quad t \geq 0 \end{cases}$$
</details>

note $f(u) = u(1-u) \Rightarrow f'(u) = 1 - 2u$. ie, equilibium $u = 0$ satisfies $f'(0) = 1$ and equilibrium $u = 1$ satisfies $f'(1) = -1$. therefore, solutions will tend towards equilibrium $u = 1$.

similar to discretization for burgers,

$$\frac{w_{ij} - w_{i,j-1}}{k} = \frac{D}{h^2}(w_{i+1,j} - 2w_{ij} + w_{i-1,j}) + w_{ik}(1 - w_{ij})$$

$$\Downarrow$$

$$(1 + 2\sigma - k(1 - w_{ij}))w_{ij} - \sigma(w_{i+1,j} + w_{i-1,j}) - w_{i,j-1} = 0$$

$$\Downarrow$$

$$F_i(z_1, \ldots, z_m) = (1 + 2\sigma - k(1 - z_i))z_i - \sigma(z_{i+1} + z_{i-1}) - w_{i,j-1} = 0, \quad z_i = w_{ij}, \text{time}$$

$$\Downarrow$$

$$\begin{aligned} F_1(z_1, \ldots, z_m) &= \frac{-3z_0 + 4z_1 - z_2}{2h} = 0 \\ F_m(z_1, \ldots, z_m) &= \frac{-z_{m-2} + 4z_{m-1} - 3z_m}{-2h} = 0 \end{aligned}$$

$$\Downarrow$$

$$DF = \begin{bmatrix} -3 & 4 & -1 & & & \\ -\sigma & 1 + 2\sigma - k + 2kz_2 & -\sigma & & & \\ & -\sigma & 1 + 2\sigma - k + 2kz_3 & -\sigma & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\sigma & 1 + 2\sigma - k + 2kz_{m-1} & -\sigma \\ & & & -1 & 4 & -3 \end{bmatrix}$$

modify eg 8.12 code for F, DF, usw.

code, matlab

↳ *1 cell hidden*

code, python

[ ] ↳ *1 cell hidden*

usw

↳ *1 cell hidden*

8.4.2 nonlinear equations in two space dimensions

like section 8.4.1, use implicit backward difference method and newton iteration but for nonlinearity also translate the mesh using the methods from 8.3.1.

example 14

apply BDM with MNI to fishers equation on unit square $[0,1] \times [0,1]$

$$\begin{cases} u_t = D\Delta u + u(1-u) \\ u(x,y,0) = 2 + cos\pi x \cdot cos\pi y, \quad 0 \leq x,y \leq 1 \\ u_{\vec{n}}(x,y,t) = 0, \quad \text{on boundary}, t \geq 0 \end{cases}$$

where D is diffusion coefficient, $u_{\vec{n}}$ is directional derivative in the outward normal condition. assume neumann (ie, no-flux) boundary conditions.

i, j represent spacial coordinates x, y. M steps along x, N steps along y with step sizes $h = \frac{x_r - x_l}{M}, k = \frac{y_t - y_b}{N}$. discretized equations at nonboundary grid points for $1 < i < m = M+1, 1 < j < n = N+1$

$$\frac{w_{ij}^t - w_{ij}^{t-\Delta t}}{\Delta t} = \frac{D}{h^2}(w_{i+1,j}^t - 2w_{ij}^t + w_{i-1,j}^t) + \frac{D}{k^2}(w_{i,j+1}^t - 2w_{ij}^t + w_{i,j-1}^t) + w_{ij}^t(1 - w_{ij}^t)$$

$$\Downarrow \quad F_{ij}(w^t) = 0$$

$$\left(\frac{1}{\Delta t} + \frac{2D}{h^2} + \frac{2D}{k^2} - 1\right) w_{ij}^t - \frac{D}{h^2} w_{i+1,j}^t - \frac{D}{h^2} w_{i-1,j}^t - \frac{D}{k^2} w_{i,j+1}^t - \frac{D}{k^2} w_{i,j-1}^t - (w_{ij}^t)^2 - \cdot$$

solve F_{ij} implicitly and use multivariate newtons bc nonlinear. flatten 2D system with $v_{i+(j-1)m} = w_{ij}$ and run it through the machine.

$$\begin{aligned}
DF_{i+(j-1)m.i+(j-1)m} &= \left(\frac{1}{\Delta t} + \frac{2D}{h^2} + \frac{2D}{k^2} - 1\right) + 2w_{ij} \\
DF_{i+(j-1)m,i+1+(j-1)m} &= -\frac{D}{h^2} \\
DF_{i+(j-1)m,i-1+(j-1)m} &= -\frac{D}{h^2} \\
DF_{i+(j-1)m,i+jm} &= -\frac{D}{k^2} \\
DF_{i+(j-1)m,i+(j-2)m} &= -\frac{D}{k^2}
\end{aligned}$$

for interior points. for points governed by neumann boundary conditions,

$$
\begin{aligned}
\text{bottom} \quad & \frac{3w_{ij} - 4w_{i,j+1} + w_{i,j+2}}{2k} = 0, \quad j = 1, 1 \le i \le m \\
\text{top} \quad & \frac{3w_{ij} - 4w_{i,j-1} + w_{i,j-2}}{2k} = 0, \quad j = n, 1 \le i \le m \\
\text{left} \quad & \frac{3w_{ij} - 4w_{i+1,j} + w_{i+2,j}}{2h} = 0, \quad i = 1, 1 \le j \le n \\
\text{right} \quad & \frac{3w_{ij} - 4w_{i-1,j} + w_{i-2,j}}{2h} = 0, \quad i = m, 1 \le j \le n
\end{aligned}
$$

$$\Downarrow$$

$$
\begin{aligned}
\text{bottom} \quad & DF_{i+(j-1)m,i+(j-1)m} = 3, \\
& DF_{i+(j-1)m,i+jm} = -4, \\
& DF_{i+(j-1)m,i+(j+1)m} = 1, \\
& b_{i+(j-1)m} = 0, \quad j = 1, 1 \le i \le m; \\
\text{top} \quad & DF_{i+(j-1)m,i+(j-1)m} = 3, \\
& DF_{i+(j-1)m,i+(j-2)m} = -4, \\
& DF_{i+(j-1)m,i+(j-3)m} = 1, \\
& b_{i+(j-1)m} = 0, \quad j = n, 1 \le i \le m; \\
\text{left} \quad & DF_{i+(j-1)m,i+(j-1)m} = 3, \\
& DF_{i+(j-1)m,i+1+(j-1)m} = -4, \\
& DF_{i+(j-1)m,i+2+(j-1)m} = 1, \\
& b_{i+(j-1)m} = 0, \quad i = 1, 1 < j < n; \\
\text{left} \quad & DF_{i+(j-1)m,i+(j-1)m} = 3, \\
& DF_{i+(j-1)m,i-1+(j-1)m} = -4, \\
& DF_{i+(j-1)m,i-2+(j-1)m} = 1, \\
& b_{i+(j-1)m} = 0, \quad i = m, 1 < j < n.
\end{aligned}
$$

› code, matlab

↳ 1 cell hidden

› code, python **not entirely soup yet**

[ ] *↳ 2 cells hidden*

usw

which goes a little something like this. ah, and this is an animation, from (a) to (b).

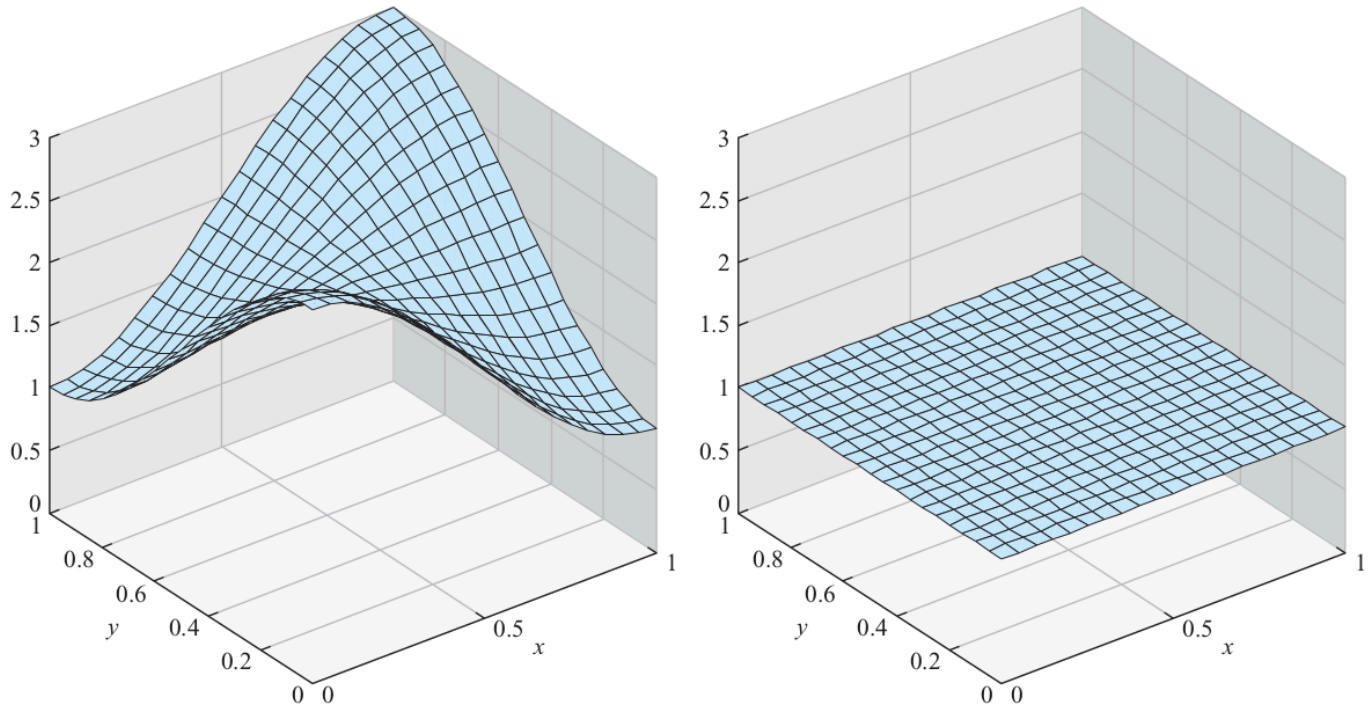

Figure 8.21 Fisher's equation with Neumann boundary conditions on a two-dimensional domain. The solution tends toward the equilibrium solution $u(x, y, t) = 1$ as t increases. (a) The initial condition $u(x, y, 0) = 2 + \cos \pi x \cos \pi y$. (b) Approximate solution after 5 time units. Step sizes $h = k = \Delta t = 0.05$.

example 15

backstory

mathematician alan turing proposed an explanation for shapes and structures found in the natural world. certain reaction-diffusion equations demonstrated emergent order and are now known as turing patterns. turing found that adding a diffusion term to a stable chemical reaction caused instability. this turing instability causes a transition where patterns might evolve into a new, spatially varying steady-state. the brusselator model (ilya prigogine link text) consists of two coupled PDEs that each represent one species of a two-species chemical reaction.

apply BDM and NMI to brusselator equation with homogeneous neumann boundary conditions on $[0, 40] \times [0, 40]$

$$\begin{cases} p_t = D_p \Delta p + p^2 q + C - (K+1)p \\ q_t = D_p \Delta q - p^2 q + Kp \\ \\ p(x,y,0) = C + 0.1 \quad 0 \le x, y \le 40 \\ q(x,y,0) = \frac{K}{C} + 0.2 \quad 0 \le x, y \le 40 \\ u_{\vec{n}}(x,y,t) = 0 \quad \text{on boundary}, t \ge 0 \end{cases}$$

ie, two coupled equations of variables p, q, diffusion coefficients D_p, D_q and parameters $C, K > 0$. equilibrium at $p \equiv C, q \equiv \frac{K}{C}$. it is known that equilibrium is stable for small K and that a turing instability occurs when

$$K > \left(1 + C\sqrt{\frac{D_p}{D_q}}\right)^2.$$

at interior points, $1 < i < m, 1 < j < n$,

$$\frac{p_{ij}^t - p_{ij}^{t-\Delta t}}{\Delta t} - \frac{D_p}{h^2}(p_{i+1,j}^t - 2p_{ij}^t + p_{i-1,j}^t) - \frac{D_p}{k^2}(p_{i,j+1}^t - 2p_{ij}^t + p_{i,j-1}^t) - (p_{ij}^t)^2 q_{ij}^t - C +$$
$$\frac{q_{ij}^t - q_{ij}^{t-\Delta t}}{\Delta t} - \frac{D_q}{h^2}(q_{i+1,j}^t - 2q_{ij}^t + q_{i-1,j}^t) - \frac{D_q}{k^2}(q_{i,j+1}^t - 2q_{ij}^t + q_{i,j-1}^t) - (q_{ij}^t)^2 q_{ij}^t - C +$$

note: this is the first example with coupled variables. also,

$$\begin{aligned} v_{i_{(j-1)m}} &= p_{ij} \quad 1 \le i \le m, 1 \le j \le n \\ v_{mn_i + (j-1)m} &= q_{ij} \quad 1 \le i \le m, 1 \le j \le n. \end{aligned}$$

› code, matlab

↳ *1 cell hidden*

› code, python

[ ] ↳ 2 cells hidden

usw

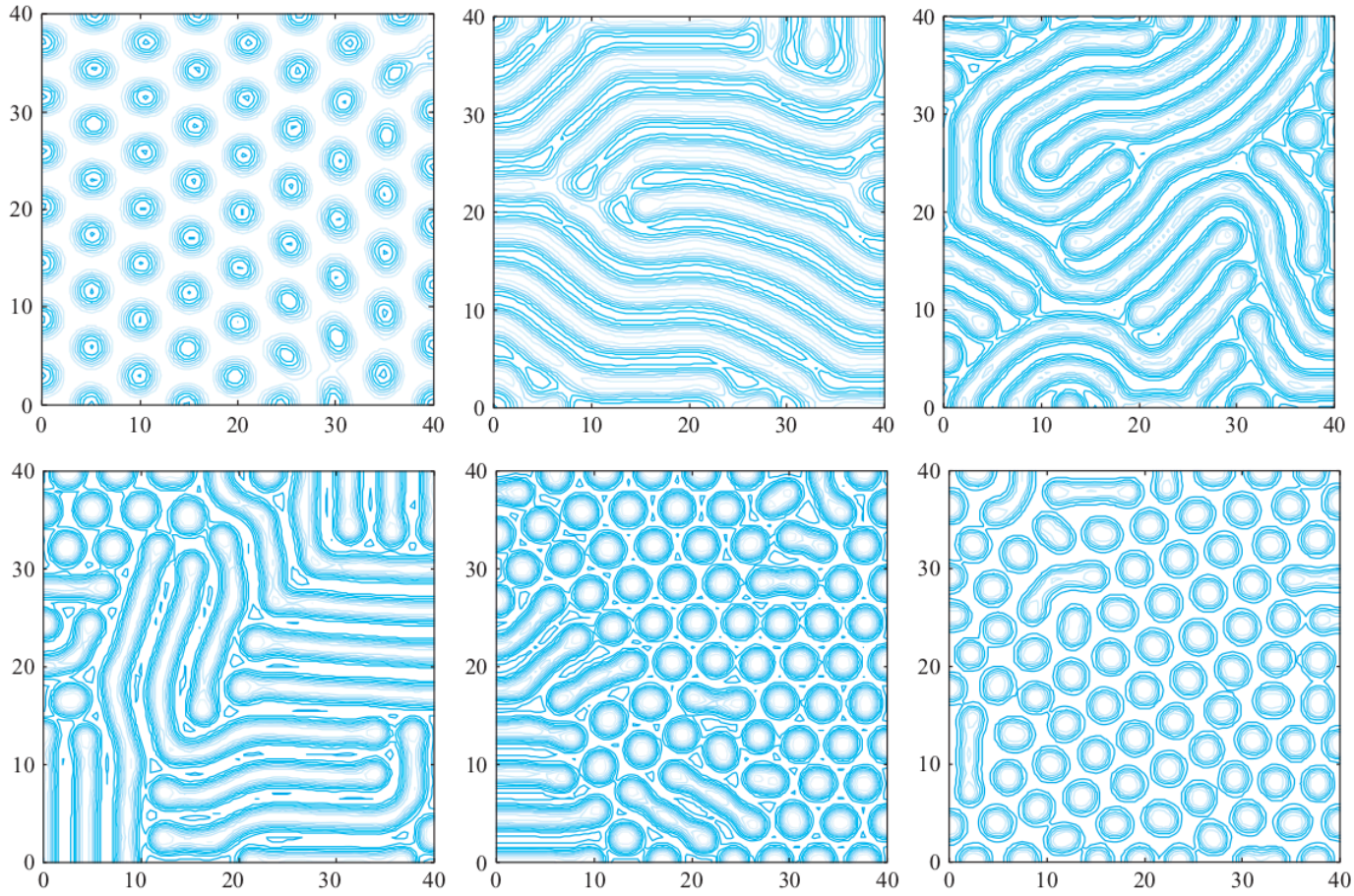

Figure 8.22 Pattern formation in the Brusselator. Contour plots of solutions $p(x, y)$ at $t = 2000$ show Turing patterns. Parameters are $D_p = 1, D_q = 8, C = 4.5$ and (a) $K = 7$ (b) $K = 8$ (c) $K = 9$ (d) $K = 10$ (e) $K = 11$ (f) $K = 12$. Settings for the finite differences are $h = k = 0.5, \Delta t = 1$.

- an alternative brusselator: alt.

you will likely need to run this in a notebook cell for the above to run.

```
pip install py-pde h5py pyfftw tqdm
```