# Solving BVPs using differentiation matrices

Robert Piché
Tampere University of Technology

March 2, 2007

In this text I present some examples to illustrate how to solve boundary value problems (BVPs) using spectral collocation *differentiation matrix* in Matlab. The examples include a BVP with Robin boundary conditions, a nonlinear BVP, an eigenvalue problem, and an initial-boundary value problem. The longest code is 16 lines. Further examples are given in [2, 5, 6].

## Spectral collocation

Consider basis functions $\phi_j$ that are polynomials of degree $N-1$ satisfying $\phi_j(x_k) = \delta_{j,k}$ for the Chebyshev nodes

$$x_k = \cos((k-1)\pi/(N-1)), \quad k = 1, \ldots, N.$$

(Note that $x_1 = 1$ and $x_N = -1$.) The polynomial

$$p(X) = \sum_{j=1}^{N} \phi_{j+1}(X) u_j$$

interpolates the points $(x_j, u_j)$, that is, $p(\mathbf{x}) = \mathbf{u}$. The values of the interpolating polynomial's $d$th derivative at the nodes are

$$p^{(d)}(\mathbf{x}) = D^{(d)}\mathbf{u},$$

where the $i, j$th element of the *differentiation matrix* $D^{(d)}$ is $\phi_j^{(d)}(x_k)$. Note that $D^{(d)} \neq (D^{(1)})^d$.

In the spectral collocation method for solving differential equations, the interpolating polynomial is required to satisfy the differential equation at the interior nodes. The values of the interpolating polynomial at the interior nodes are $p(\mathbf{x}_{2:N-1}) = \mathbf{u}_{2:N-1} = I_{2:N-1,:}\mathbf{u}$ and the derivative values are $p^{(d)}(\mathbf{x}_{2:N-1}) = D^{(d)}_{2:N-1,:}\mathbf{u}$. Boundary conditions that involve the derivative can be handled by using the formulas

$$p^{(d)}(1) = D^{(d)}_{1,:}\mathbf{u}, \quad p^{(d)}(-1) = D^{(d)}_{N,:}\mathbf{u}$$

The Differentiation Matrix Suite [6] provides two useful Matlab functions for spectral collocation, `chebdif` and `chebint`. The function call `[x,D]=chebdif(N,M)` computes the differentiation matrices for $d = 1, 2, \ldots, M$, where $0 < M \le N - 1$. The subarray `D(:,:,d)` contains the $N \times N$ matrix $D^{(d)}$. The column vector `x` contains the Chebyshev nodes with $\mathbf{x}_1 = 1$ and $\mathbf{x}_N = -1$.

The function call `p=chebint(u,X)` evaluates the polynomial that interpolates the data vector `u` at the Chebyshev nodes. The polynomial is evaluated at the values given in vector `X`.

# 1 Linear BVP with Robin boundary condition

The problem

$$y'' - 2xy' + 2y = 4e^{x^2}, \quad 2y(1) - y'(1) = 1,\ 2y(-1) + y'(-1) = -1$$

is used in [6] to illustrate the solution of a boundary value problem with Robin boundary conditions. Here I present an alternative (simpler) solution.

For the interpolating polynomial to satisfy the differential equation at each interior node, the collocation equation

$$p''(\mathbf{x}_{2:N-1}) + Qp'(\mathbf{x}_{2:N-1}) + 2p(\mathbf{x}_{2:N-1}) = \mathbf{f}$$

should be satisfied, where $Q = \mathrm{diag}(-2(\mathbf{x}_{2:N-1}))$ and $\mathbf{f} = 4e^{\mathbf{x}^2_{2:N-1}}$. Substituting the differentiation matrix relations, the collocation equation can be written as

$$\left(D^{(2)}_{2:N-1,:} + QD^{(1)}_{2:N-1,:} + 2I_{2:N-1,:}\right)\mathbf{u} = \mathbf{f}$$

The boundary conditions $2p(1) - p'(1) = 1$ and $2p(-1) + p'(-1) = -1$ are satisfied when

$$(2I_{1,:} - D^{(1)}_{1,:})\mathbf{u} = 1, \quad (2I_{N,:} + D^{(1)}_{N,:})\mathbf{u} = -1$$

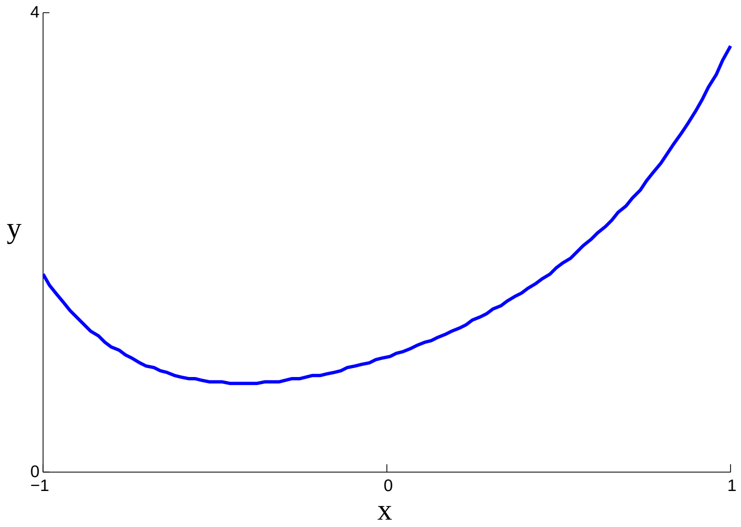
The equations can be collected into the single linear matrix equation

$$\begin{bmatrix} D^{(2)}_{2:N-1,:} + QD^{(1)}_{2:N-1,:} + 2I_{2:N-1,:} \\ 2I_{1,:} - D^{(1)}_{1,:} \\ 2I_{N,:} + D^{(1)}_{N,:} \end{bmatrix} \mathbf{u} = \begin{bmatrix} \mathbf{f} \\ 1 \\ -1 \end{bmatrix}$$

The following code computes the BVP solution at the nodes, interpolates it with `chebint`, and draws a plot:

```
N=10;
[X,D]=chebdif(N,2);
i=2:N-1;   % inner nodes
Q=diag(-2*X(i));
f=4*exp(X(i).^2);
I=eye(N);
u=[D(i,:,2)+Q*D(i,:,1)+2*I(i,:)
   2*I(1,:)-D(1,:,1)
   2*I(N,:)+D(N,:,1)]\[f;1;-1];
XX=linspace(-1,1)';
plot(XX,chebint(u,XX))
```

The solution curve is:




**Exercise 1** Solve the differential equation $u''+u=1$ on $x \in (0,1)$ with boundary conditions $u(0)=u'(1)=0$. Verify your answer using the exact solution $u(x)=1-\cos(x)-\sin(x)\tan(1)$.

**Exercise 2** Solve the differential equation $-(e^{x/10}u')'=1$ on $x \in (0,1)$ with boundary conditions $u(0)=u'(1)=0$. Verify your answer using the exact solution $u(x)=(10x+90)e^{-x/10}-90$.

# 2 Nonlinear ODE

The boundary value problem

$$2xy''+y'-y^3=0, \quad y(0)=\frac{1}{10},\ y(16)=\frac{1}{6}.$$

is presented in [3, example 7] as an example of a problem that the Matlab solver `bvp4c` cannot handle directly because of the singularity at $x=0$. The spectral collocation method can be used directly because it does not evaluate the ODE at the end points. This example also illustrates the solution of nonlinear BVPs using Newton iteration.

The change of variables $Y(X)=y(8X+8)$ transforms the boundary value problem to

$$\frac{1}{4}(X+1)Y''+\frac{1}{8}Y'-Y^3=0, \quad Y(-1)=\frac{1}{10},\ Y(1)=\frac{1}{6}$$

For the interpolating polynomial to satisfy the differential equation at each interior node, the collocation equation

$$Qp''(\mathbf{x}_{2:N-1})+\frac{1}{8}p'(\mathbf{x}_{2:N-1})-f(p(\mathbf{x}_{2:N-1}))=0$$

should be satisfied, where $Q = \frac{1}{4}\mathrm{diag}(\mathbf{x}_{2:N-1} + 1)$ and $f(u) = u^3$. Substituting the differentiation matrix relations, the collocation equation can be written as

$$QD^{(2)}_{2:N-1,:}\mathbf{u} + \frac{1}{8}D^{(1)}_{2:N-1,:}\mathbf{u} - I_{2:N-1,:}f(\mathbf{u}) = 0$$

The boundary conditions are modelled by the equations

$$\mathbf{u}_N - \frac{1}{10} = 0, \quad \mathbf{u}_1 - \frac{1}{6} = 0$$

The solution vector $\mathbf{u}$ can be found by solving the nonlinear equation system

$$\mathbf{F}(\mathbf{u}) = \begin{bmatrix} QD^{(2)}_{2:N-1,:}\mathbf{u} + \frac{1}{8}D^{(1)}_{2:N-1,:}\mathbf{u} - I_{2:N-1,:}f(\mathbf{u}) \\ \mathbf{u}_N - \frac{1}{10} \\ \mathbf{u}_1 - \frac{1}{6} \end{bmatrix} = \mathbf{0}$$

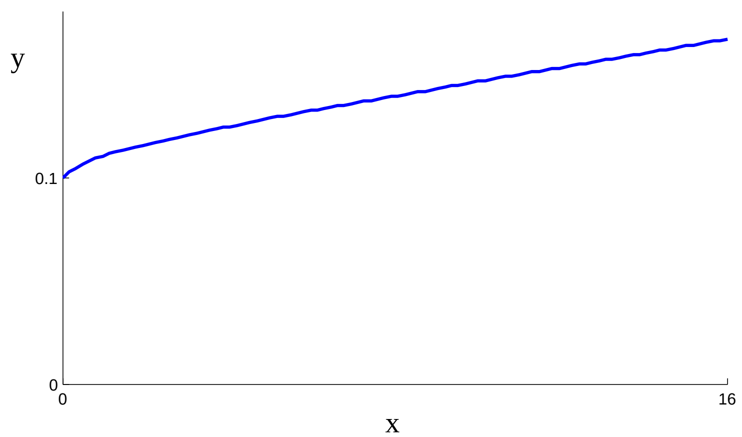
with Newton's iterative method and the initial iterate $\mathbf{u} = \mathbf{0}$. The jacobian is

$$\partial\mathbf{F}/\partial\mathbf{u} = \begin{bmatrix} QD^{(2)}_{2:N-1,:} + \frac{1}{8}D^{(1)}_{2:N-1,:} - I_{2:N-1,:}\mathrm{diag}(f'(\mathbf{u})) \\ I_{N,:} \\ I_{1,:} \end{bmatrix}$$

The interpolating polynomial can then be evaluated using `chebint`. The code

```
N=10;
[X,D]=chebdif(N,2);
i=2:N-1;
Q=diag((X(i)+1)/4);
I=eye(N);
u=zeros(N,1);
for k=1:3  % Newton iterations
    F=[(Q*D(i,:,2)+D(i,:,1)/8)*u-I(i,:)*u.^3;u(N)-1/10;u(1)-1/6];
    dF=[Q*D(i,:,2)+D(i,:,1)/8-I(i,:)*diag(3*u.^2);I(N,:);I(1,:)];
    u=u-dF\F;
end
XX=linspace(-1,1)';
plot(8*XX+8,chebint(u,XX))
```

plots this curve:



**Exercise 3** Find two solutions of the Bratu equation $u'' + e^u = 0$ with boundary conditions $u(0) = u(1) = 0$.

**Exercise 4** Solve the equation $y'' + \frac{2}{x}y' = \frac{100y}{10y+1}$ with boundary conditions $y'(0) = 0$, $y(1) = 1$.

# 3 Eigenvalue problem

Latzko's equation is given in [4, p.143] as

$$((1 - x^7)y')' + \lambda x^7 y = 0$$

with boundary conditions $y(0) = 0$ and $y'(1) = \frac{1}{7}\lambda y(1)$. The ODE is singular at the endpoint $x = 1$. The task is to compute eigenvalues. The Matlab solver `bvp4c` cannot directly handle the singularity, and eigenvalues have to be treated as parameters to be found by solving nonlinear equations. The spectral collocation method can treat this problem directly and the `eig` function can be used to find eigenvalues.

The change of variables $Y(X) = y((X + 1)/2)$ transforms the boundary value problem to

$$Y'' + qY' + \lambda rY = 0, \quad Y(-1) = 0,\ 2Y'(1) = \tfrac{1}{7}\lambda Y(1),$$

where

$$q(X) = -\frac{\frac{7}{2}(X/2 + 1/2)^6}{1 - (X/2 + 1/2)^7}, \quad r(X) = \frac{\frac{1}{4}(X/2 + 1/2)^7}{1 - (X/2 + 1/2)^7}.$$

The collocation equation is

$$p''(\mathbf{x}_{2:N-1}) + Qp'(\mathbf{x}_{2:N-1}) + \lambda Rp(\mathbf{x}_{2:N-1}) = 0,$$

where $Q = \mathrm{diag}(q(\mathbf{x}_{2:N-1}))$ and $R = \mathrm{diag}(r(\mathbf{x}_{2:N-1}))$. Substituting the differentiation matrix relations gives

$$(D^{(2)}_{2:N-1,:} + QD^{(1)}_{2:N-1,:} + \lambda RI_{2:N-1,:})\mathbf{u} = 0.$$

The boundary conditions $p(-1) = 0$ and $2p'(1) = \frac{1}{7}\lambda p(1)$ are modelled by the equations

$$I_{N,:}\mathbf{u} = \lambda \mathbf{0}^T\mathbf{u}, \quad 2D^{(1)}_{1,:}\mathbf{u} = \tfrac{1}{7}\lambda I_{1,:}\mathbf{u}.$$

Combining these equations gives the generalized eigenvalue problem

$$\begin{bmatrix} -D^{(2)}_{2:N-1,:} - QD^{(1)}_{2:N-1,:} \\ I_{N,:} \\ 2D^{(1)}_{1,:} \end{bmatrix} \mathbf{u} = \lambda \begin{bmatrix} RI_{2:N-1,:} \\ \mathbf{0}^T \\ \frac{1}{7}I_{1,:} \end{bmatrix} \mathbf{u}.$$

The code

```
N=10;
[X,D]=chebdif(N,2);
i=2:N-1;
x=X(i)/2+1/2;
Q=diag(-(7/2)*x.^6./(1-x.^7));
R=diag(x.^7./(1-x.^7)/4);
I=eye(N);
A=[-D(i,:,2)-Q*D(i,:,1); I(N,:); 2*D(1,:,1)];
B=[R*I(i,:); zeros(1,N); I(1,:)/7];
lambda=eig(A,B)
```

produces

```
lambda =

   5.3126e+13
   6.6264e+08
   1.5248e+06
        32967
         2878
       727.78
       421.48
       8.7275
       152.47
       Inf
```

The three smallest eigenvalues agree reasonably well with the values 8.728, 152.45, 435.2 given in [4].

**Exercise 5** Find the first three positive eigenvalues of the Mathieu equation $y'' + \lambda y = 10y\cos(2x)$ with boundary conditions $y'(0) = 0$, $y'(\pi) = 0$.

**Exercise 6** The natural frequencies of vibration of the radial modes of a circular membrane are the eigenvalues of the Bessel equation $x^2y'' + xy + \mu^2x^2y = 0$ with boundary conditions $y'(0) = y(1) = 0$. Find the first three positive eigenvalues and compare to the values of the zeros of $J_0$ listed in [1] or at
`http://mathworld.wolfram.com/BesselFunctionZeros.html`

# 4 Initial-boundary value problem

A Black-Scholes equation for pricing a vanilla European call option is

$$\dot{y} = -\tfrac{1}{2}\sigma^2 y'' - (r - \tfrac{1}{2}\sigma^2)y' + ry$$

where the dot represents time differentiation and the prime represents differentiation with respect to logarithmic price $x = \log(S)$. The final condition (i.e. the option's value in one year) is $y(x,1) = \max(e^x - K, 0)$ and the boundary conditions are $y(-L,t) = 0$ and $y(L,t) = e^L - Ke^{-r(T-t)}$, where the value $L = \log(200)$ is used to approximate infinity. The task is to find today's option value $y(x,0)$ for $\sigma = 0.2$, $r = 0.1$, and $K = 100$. This problem can easily be solved using the Matlab function `pdepe`, but here it is solved using the spectral collocation method.

The change of variables $Y(X,t) = y(XL,t)$ transforms the boundary value problem to

$$\dot{Y} = -\frac{\sigma^2}{2L^2}Y'' - \frac{1}{L}(r - \tfrac{1}{2}\sigma^2)Y' + rY$$

The differentiation matrix model is

$$I_{2:N-1,:}\dot{\mathbf{u}} = -\frac{\sigma^2}{2L^2}D^{(2)}_{2:N-1,:}\mathbf{u} - \frac{1}{L}(r - \tfrac{1}{2}\sigma^2)D^{(1)}_{2:N-1,:}\mathbf{u} + rI_{2:N-1,:}\mathbf{u}$$

The boundary conditions are modelled as

$$\mathbf{0}^T\dot{\mathbf{u}} = I_{N,:}\mathbf{u}, \quad \mathbf{0}^T\dot{\mathbf{u}} = e^L - Ke^{-r(T-t)} - I_{1,:}\mathbf{u}$$

These equations can be collected to give the differential algebraic system

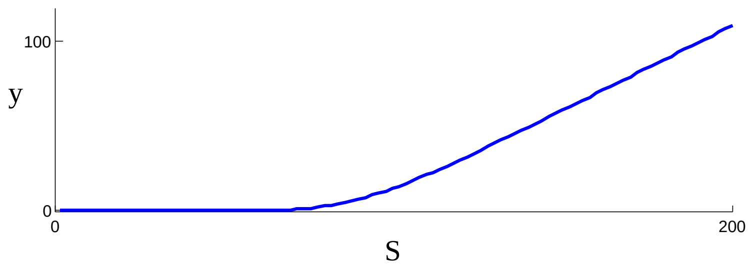
$$\begin{bmatrix} I_{2:N-1,:} \\ \mathbf{0}^T \\ \mathbf{0}^T \end{bmatrix} \dot{\mathbf{u}} = \begin{bmatrix} -\frac{\sigma^2}{2L^2}D^{(2)}_{2:N-1,:} - \frac{1}{L}(r - \frac{1}{2}\sigma^2)D^{(1)}_{2:N-1,:} + rI_{2:N-1,:} \\ I_{N,:} \\ -I_{1,:} \end{bmatrix} \mathbf{u} + \begin{bmatrix} \mathbf{0} \\ 0 \\ e^L - Ke^{-r(T-t)} \end{bmatrix}$$

with the terminal condition $\mathbf{u}(1) = \max(e^{L\mathbf{x}} - K, 0)$.

The stiff solver `ode15s` can handle DAE problems with backward time evolution. The solution can be evaluated using `chebint`. The m-file

```
K=100; sigma=0.2; r=0.1; S0=200; T=1;
N=10;
[X,D]=chebdif(N,2);
i=2:N-1;
L=log(S0);
I=eye(N);
A=[-sigma^2/L^2/2*D(i,:,2)-(r-sigma^2/2)/L*D(i,:,1)+r*I(i,:);
    I(N,:); -I(1,:)];
M=[I(i,:);zeros(2,N)];
op=odeset('jacobian',A,'mass',M,'abstol',1e-5,'reltol',1e-4);
dudt=@(t,u,K,r,S0,T,A) A*u+[zeros(length(u)-1,1);S0-K*exp(-r*(T-t))];
u0=max(exp(L*X)-K,0);
[t,u]=ode15s(dudt,[T,0],u0,op,K,r,S0,T,A);
S=linspace(1,S0)';
y=chebint(u(end,:)',log(S)/L);
plot(S,y)
```

plots this curve:




**Exercise 7** Compare the solutions computed above with the analytical solution

$$y = S\,\mathrm{N}(d_1) - Ke^{-r(T-t)}\mathrm{N}(d_2),$$

where

$$d_1 = \frac{\ln(S/K) + (r + \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}}, \quad d_2 = d_1 - \sigma\sqrt{T-t},$$

and N is the standard normal cumulative distribution function

$$\mathrm{N}(d) = \frac{1}{2} + \frac{1}{2}\mathrm{erf}(d/\sqrt{2}).$$

**Exercise 8** Solve the one-dimensional heat equation $u_t = u_{xx} + h(x)$ where $h$ is the `humps` function, the initial condition is $u(x, 0) = 0$, and the boundary conditions are $u(0, t) = 0$, $u(1, t) = 0$. Plot the solution at $t = 0.01$.

# References

[1] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover, 1964, online at `http://www.convertit.com/Go/ConvertIt/Reference/AMS55.ASP`

[2] R. Piché and J. Kanniainen, Solving financial differential equations using differentiation matrices, World Congress of Engineering, London, 2007. `http://alpha.cc.tut.fi/~piche/finance/2007A/Report.pdf`

[3] L. F. Shampine, J. Kierzenka, M. W. Reichelt (2000), Solving Boundary Value Problems for Ordinary Differential Equations in MATLAB with `bvp4c`, `http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?ref=bvp_tutorial&objectId=3819&objectType=file`

[4] L. F. Shampine, I. Gladwell, S. Thompson, *Solving ODEs with MATLAB*, Cambridge University Press, 2003.

[5] L. N. Trefethen, *Spectral Methods in Matlab*, SIAM, 2000.

[6] J. A. C. Weideman, S. C. Reddy, A MATLAB Differentiation Matrix Suite, *ACM Transactions on Mathematical Software*, Vol. 26, No. 4, Dec. 2000, p. 465–519. The codes are available at `http://dip.sun.ac.za/~weideman/research/differ.html`