Fourier spectral methods with Dirichlet/Neumann BCs

- For PDEs with reflection symmetry in x ($\leftrightarrow$ only even x-derivs for a single-variable PDE)

Example: heat eqn:

$$q_t = a q_{xx} \quad , \quad 0 < x < L$$

$$\left.\begin{aligned} q(0,t) &= 0 \\ q(L,t) &= 0 \end{aligned}\right\} \text{Dirichlet BCs} \qquad q(\alpha,t) = 0 \text{ or } q_x(\alpha,t) = 0, \quad \alpha = 0, L.$$

$$q(x,0) = q_0(x)$$

Strategy: Extension to a periodic function

Odd extn: Dirichlet
Even extn: Neumann

Let
$$\bar{q}(x,t) = \begin{cases} q(x,t) & , \ 0 \le x < L \\ -q(2L-x,t) & , \ L < x < 2L \\ q(x \bmod 2L, t) & , \ \text{other } x \end{cases}$$

and $\bar{q}, \bar{q}_x$ conts. at $x = 0, L$

Since $-q(2L-x,t)$ is also a soln to PDE, $\bar{q}(x,t)$ is soln over extended domain. $\Rightarrow$ Solve

$$\bar{q}_t = a\bar{q}_{xx} \qquad 0 < x < 2L$$

· periodic BCs

$$\bar{q}(x,0) = \begin{cases} q_0(x) & 0 \le x \le L \\ -q_0(2L-x) & L < x < 2L. \end{cases}$$

For Neumann BC's

$$\bar{q}(x,t) = \begin{cases} q(x,t) & 0 \le x < L \\ \bar{q}(2L-x,t) & L < x < 2L \end{cases}$$

+ same procedure

```
% FS_heat :  Fourier Spectral method +RK4 on heat eqn with Dirichlet BCs
%   q_t = a*q_xx   0 < x < L=1, a = 1, 0 < t < T = 0.05
%   q(x,0) = exp(-(x - 0.5).^2/(2*sigma^2)), sigma = 0.1
%   BCs:   Dirichlet  q(1,t) = q(0,t) = 0
% Use N = 2^p modes on an extended periodic domain [0,2]
  p = 3;
  nu = 0.2;  % nu = a*dt/dx^2;  nondimensional timestep dt; RK4 stability limit nu < 2.8/pi^2 = 0.28
  nx = 2^p;
  dx = 2*L/nx;
  dt = nu*dx^2/a;
  nt = round(T/dt);
  x = dx*(0:(nx/2-1));
  q0 = exp(-(x - 0.5).^2/(2*sigma^2));  % Initial condition at gridpoints
  qd0 = [0 q0(2:nx/2) 0 -q0(nx/2:-1:2)]; % Odd periodic extn
  xx = dx*(0:(nx-1));  % Extended periodic grid
  k = [0:(nx/2-1) -nx/2:-1]*2*pi/(2*L); % Wavenumbers on extended grid
  qdhat = fft(qd0);
  for it = 1:nt
    % RK4 step for Fourier coeffs of Dirichlet solution
    d1 = -dt*a*k.^2.*qdhat;
    d2 = -dt*a*k.^2.*(qdhat + 0.5*d1);
    d3 = -dt*a*k.^2.*(qdhat + 0.5*d2);
    d4 = -dt*a*k.^2.*(qdhat + d3);
    qdhat = qdhat + (d1 + 2*d2 + 2*d3 + d4)/6; % New qdhat
   end
   qd = real(ifft(qdhat));
```