# Understanding 2D heat equation code

## files in .../Kernels

- k_n1n1_uu.m
- k_n1n1_uv.m
- k_n1n1_vv.m
- k_n1n_u3.m
- k_n1n_v3.m
- k_nn_11.m
- k_nn_31.m
- k_nn_33.m
- k_nn_u1.m
- k_nn_u3.m
- k_nn_uu.m
- k_nn_uv.m
- k_nn_v1.m

### ***files starting with k are covariance functions:***

k.m = $k_{u,u}^{n+1,n+1}$

k_n1n*v3.m = $k^{n+1,n}*{v,3}$

...

There are corresponding to the functions in the Raissi's paper "Numerical Gaussian Processes for Time-dependent and Non-linear Partial Differential Equations"

$$
\begin{aligned}
k_{u,v}^{n+1,n+1} &= \frac{d}{dx_2'} k_{u,u}^{n+1,n+1}, \qquad (54)\\
k_{u,3}^{n+1,n} &= k_{u,u}^{n+1,n+1} - \frac{1}{2}\Delta t \frac{d^2}{dx_1'^2} k_{u,u}^{n+1,n+1} - \frac{1}{2}\Delta t \frac{d^2}{dx_2'^2} k_{u,u}^{n+1,n+1},\\
k_{v,v}^{n+1,n+1} &= \frac{d}{dx_2}\frac{d}{dx_2'} k_{u,u}^{n+1,n+1},\\
k_{v,3}^{n+1,n} &= \frac{d}{dx_2} k_{u,u}^{n+1,n+1} - \frac{1}{2}\Delta t \frac{d}{dx_2}\frac{d^2}{dx_1'^2} k_{u,u}^{n+1,n+1} - \frac{1}{2}\Delta t \frac{d}{dx_2}\frac{d^2}{dx_2'^2} k_{u,u}^{n+1,n+1},\\
k_{u,v}^{n,n} &= \frac{d}{dx_2'} k_{u,u}^{n,n},\\
k_{u,3}^{n,n} &= -\frac{1}{2}\Delta t \frac{d^2}{dx_1'^2} k_{u,u}^{n,n} - \frac{1}{2}\Delta t \frac{d^2}{dx_2'^2} k_{u,u}^{n,n},\\
k_{u,1}^{n,n} &= k_{u,u}^{n,n}\\
k_{v,v}^{n,n} &= \frac{d}{dx_2}\frac{d}{dx_2'} k_{u,u}^{n,n},\\
k_{v,3}^{n,n} &= -\frac{1}{2}\Delta t \frac{d}{dx_2}\frac{d^2}{dx_1'^2} k_{u,u}^{n,n} - \frac{1}{2}\Delta t \frac{d}{dx_2}\frac{d^2}{dx_2'^2} k_{u,u}^{n,n},\\
k_{v,1}^{n,n} &= \frac{d}{dx_2} k_{u,u}^{n,n},
\end{aligned}
$$

***files starting with D are differentiation operators:***

D3kDx2Dy1s.m = $\frac{d}{dx_2}\frac{d^2}{x_1'^2}$

where 3 denotes third derivative, k is the covariance functions which is inserted, x2 denotes $x_2$ , y denotes x', and s means the square of the variable.

Another example:

DskDx2Dy2.m = $\frac{d}{dx_2}\frac{d}{x_2'}$

## files in the main repository

### likelihood.m

It is the definition of the NLML function of 2D heat equation.

In the paper, it looks like

$$\begin{bmatrix} \boldsymbol{u}_D^{n+1} \\ \boldsymbol{v}_N^{n+1} \\ \boldsymbol{u}_D^n \\ \boldsymbol{v}_N^n \\ \boldsymbol{u}_3^n \\ \boldsymbol{u}_1^n \end{bmatrix} \sim \mathcal{N}\left(0, \boldsymbol{K}\right),$$

The matrix $\boldsymbol{K}$ used in the distribution (44) is given by

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{D,D}^{n+1,n+1} & \boldsymbol{K}_{D,N}^{n+1,n+1} & 0 & 0 & \boldsymbol{K}_{D,3}^{n+1,n} & 0 \\ & \boldsymbol{K}_{N,N}^{n+1,n+1} & 0 & 0 & \boldsymbol{K}_{N,3}^{n+1,n} & 0 \\ & & \boldsymbol{K}_{D,D}^{n,n} & \boldsymbol{K}_{D,N}^{n,n} & \boldsymbol{K}_{D,3}^{n,n} & \boldsymbol{K}_{D,1}^{n,n} \\ & & & \boldsymbol{K}_{N,N}^{n,n} & \boldsymbol{K}_{N,3}^{n,n} & \boldsymbol{K}_{N,1}^{n,n} \\ & & & & \boldsymbol{K}_{3,3}^{n,n} & \boldsymbol{K}_{3,1}^{n,n} \\ & & & & & \boldsymbol{K}_{1,1}^{n,n} \end{bmatrix}.$$

where D denotes Dirichlet boundary conditions:

$$u(t,0,x_2) = u(t,1,x_2) = 0, \quad u(t,x_1,0) = 0,$$

while N corresponds to a Neumann-type boundary condition:

$$u_{x_2}(t,x_1,1) = 0.$$

In the code, each K is assigned corresponding covariance function to obtain the matrix K:

```
K_n1n1_DD = k_n1n1_uu(x_D, x_D, hyp(1:end-1), 0) + eye(N_D).*jitter;
K_n1n1_DN = k_n1n1_uv(x_D, x_N, hyp(1:end-1), 0);
K_n1n_D3 = k_n1n_u3(x_D, x_u, hyp(1:end-1), 0);
K_n1n1_NN = k_n1n1_vv(x_N, x_N, hyp(1:end-1), 0) + eye(N_N).*jitter;
K_n1n_N3 = k_n1n_v3(x_N, x_u, hyp(1:end-1), 0);

K_nn_DD = k_nn_uu(x_D, x_D, hyp(1:end-1), 0) + eye(N_D).*jitter;
K_nn_DN = k_nn_uv(x_D, x_N, hyp(1:end-1), 0);
K_nn_D3 = k_nn_u3(x_D, x_u, hyp(1:end-1), 0);
K_nn_D1 = k_nn_u1(x_D, x_u, hyp(1:end-1), 0);
K_nn_NN = k_nn_vv(x_N, x_N, hyp(1:end-1), 0) + eye(N_N).*jitter;
K_nn_N3 = k_nn_v3(x_N, x_u, hyp(1:end-1), 0);
K_nn_N1 = k_nn_v1(x_N, x_u, hyp(1:end-1), 0);

K_nn_33 = k_nn_33(x_u, x_u, hyp(1:end-1), 0) + eye(N_u).*sigma_u + eye(N_u).*jitter;
K_nn_31 = k_nn_31(x_u, x_u, hyp(1:end-1), 0);
K_nn_11 = k_nn_11(x_u, x_u, hyp(1:end-1), 0) + eye(N_u).*sigma_u + eye(N_u).*jitter;

K = [K_n1n1_DD K_n1n1_DN zeros(N_D,N_D) zeros(N_D,N_N) K_n1n_D3 zeros(N_D,N_u);
     K_n1n1_DN' K_n1n1_NN zeros(N_N,N_D) zeros(N_N,N_N) K_n1n_N3 zeros(N_N,N_u);
     zeros(N_D,N_D) zeros(N_D,N_N) K_nn_DD  K_nn_DN  K_nn_D3 K_nn_D1;
     zeros(N_N,N_D) zeros(N_N,N_N) K_nn_DN' K_nn_NN  K_nn_N3 K_nn_N1;
     K_n1n_D3' K_n1n_N3' K_nn_D3' K_nn_N3' K_nn_33 K_nn_31;
     zeros(N_u,N_D) zeros(N_u,N_N) K_nn_D1' K_nn_N1' K_nn_31' K_nn_11];
```