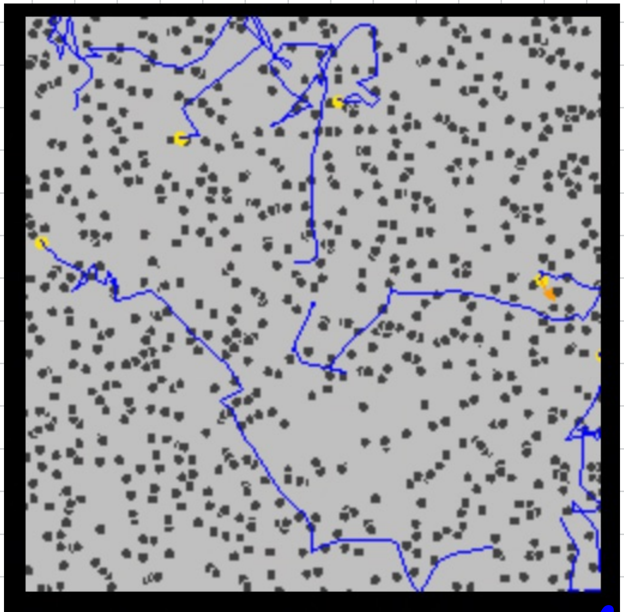
https://peterroelants.github.io/posts/gaussian-process-tutorial/

What is a Gaussian Process?

- stochastic process of random variables
- Gaussian distribution

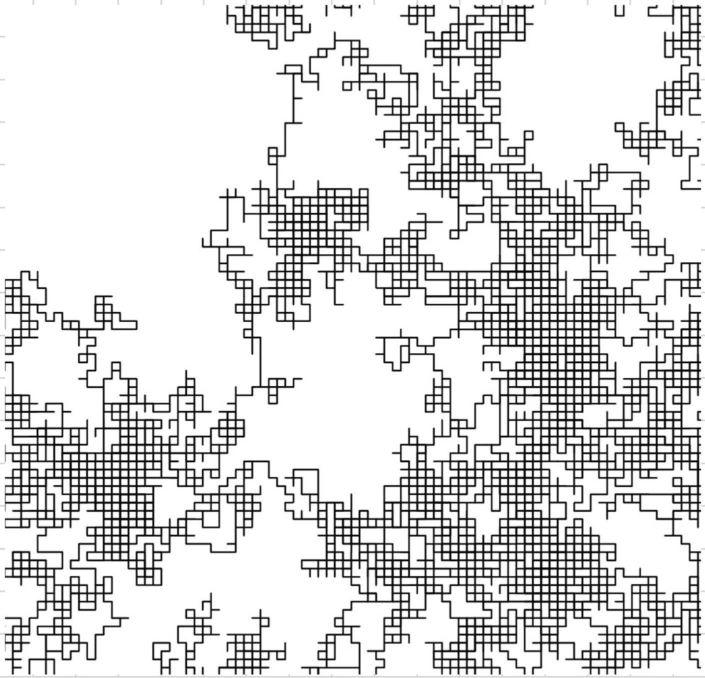
random change over time due to uncertainty

Brownian motion



random movement of particles in a suspended liquid

(lattice) Random walk



Consider random process

$d(t)$

so that

$$d(t+\Delta t) = d(t) + \Delta d$$

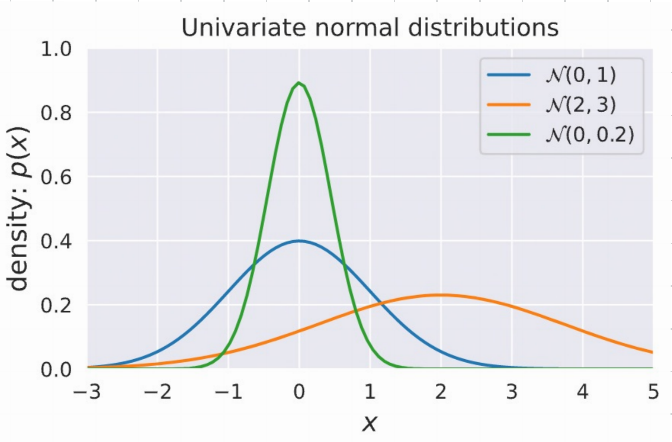
$$\Delta d \sim N(0, \Delta t)$$

μ, mean 0

variance $\sigma^2 = \Delta t$

continuous probability distribution:

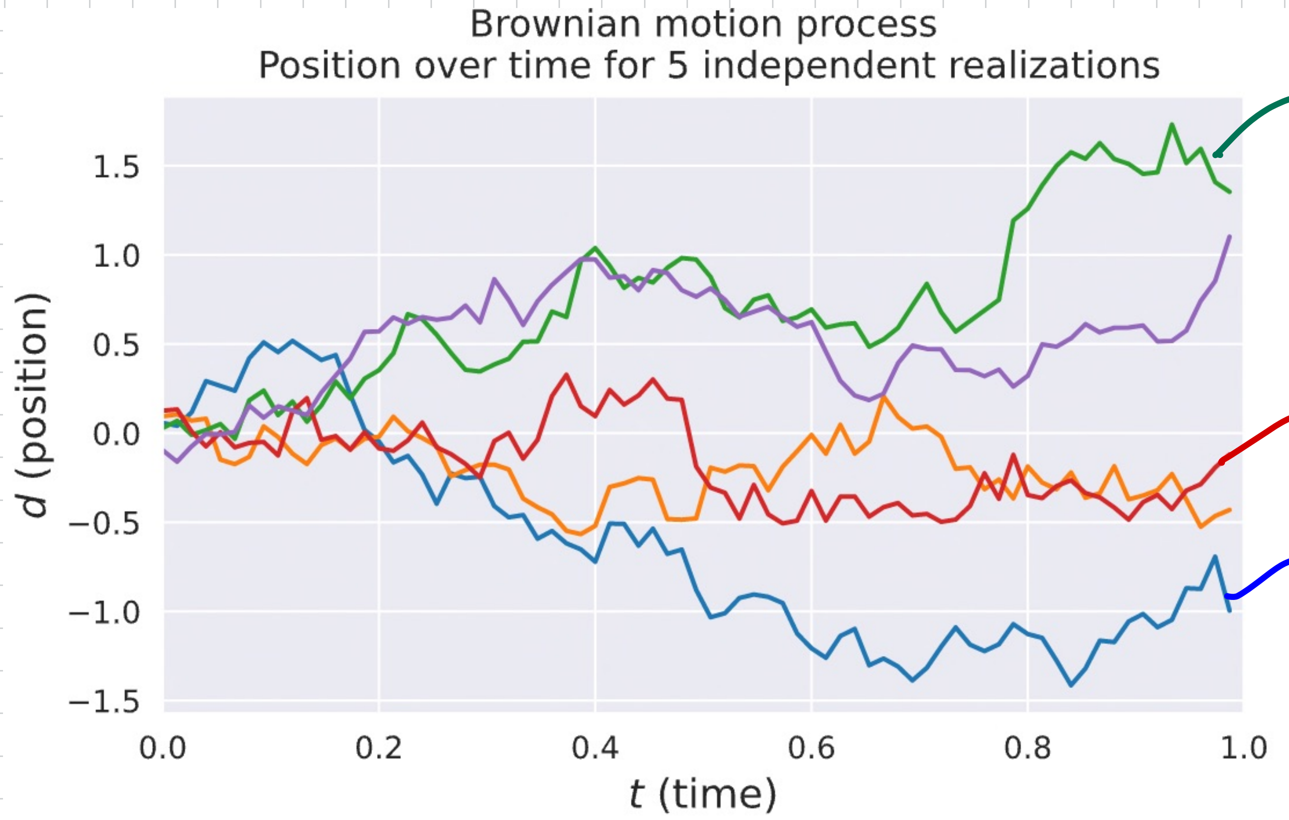
$$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



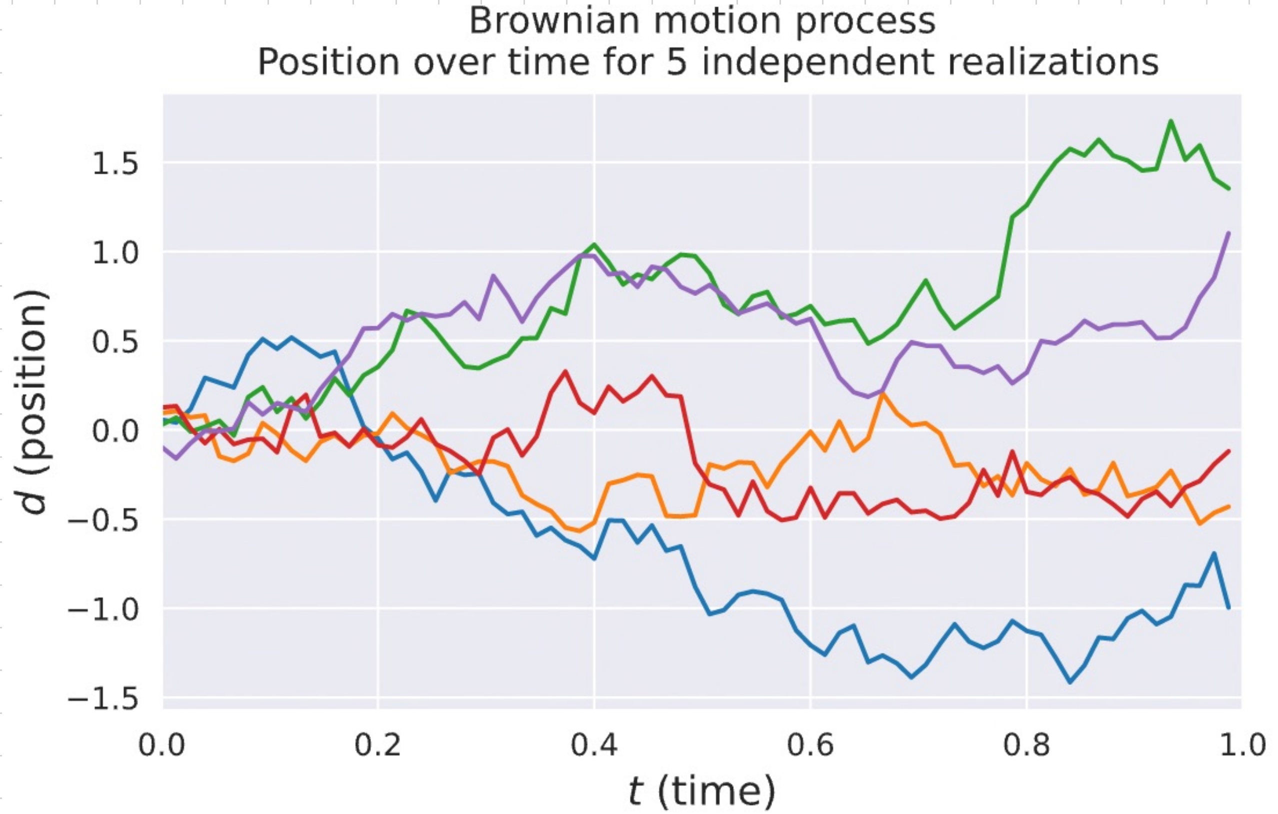
1D <u>example</u>

move particle Δt and distance Δd

$\Delta d \sim N(0, \Delta t)$



→ random distribution over functions


Brownian motion process
Position over time for 5 independent realizations
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
d (position)
0.0
0.2
0.4
0.6
0.8
1.0
t (time)

GP

defined by

$m(x)$ = mean functions

$k(x,x')$ = covariance function

we say that

$$f(x) \sim GP(m(x), k(x,x'))$$

Over any subset $\{x_1 \ldots\ldots x_n\}$

we have a Gaussian distribution

$$f(\underbrace{\{x_1 \ldots x_n\}}_{X}) \sim N(m(X), k(X,X))$$

multivariate Gaussian has joint probabality:

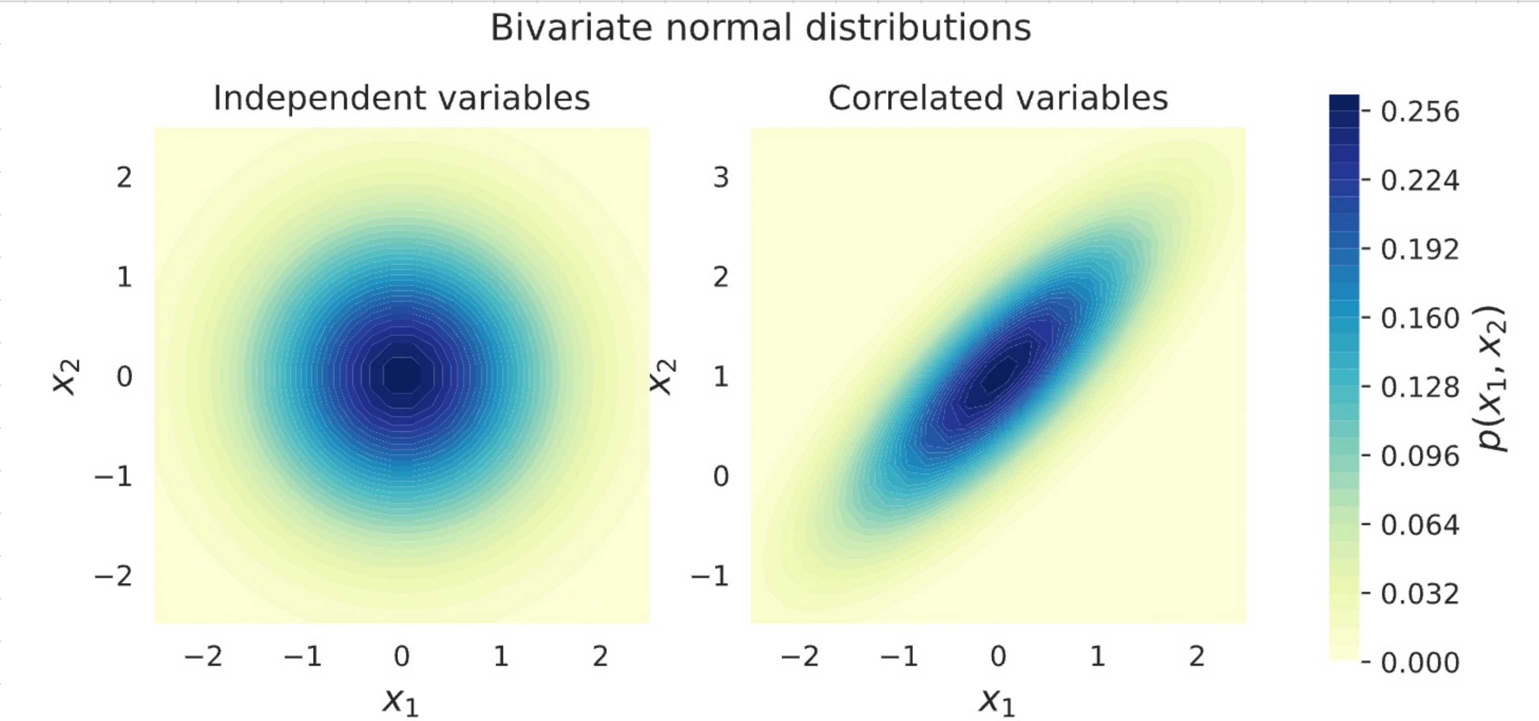
$$P(\underline{x} \mid \mu, \Sigma)$$

r.v. x d

mean

covariance matrix: dependence of any two random variable,

$$= \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} e^{-\frac{1}{2}\left((\underline{x}-\mu)^T \Sigma^{-1} (\underline{x}-\mu)\right)}$$



$N\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right)$

$N\left(\begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1 & 0.8\\0.8 & 1\end{bmatrix}\right)$

→ an increase in the prob. of x

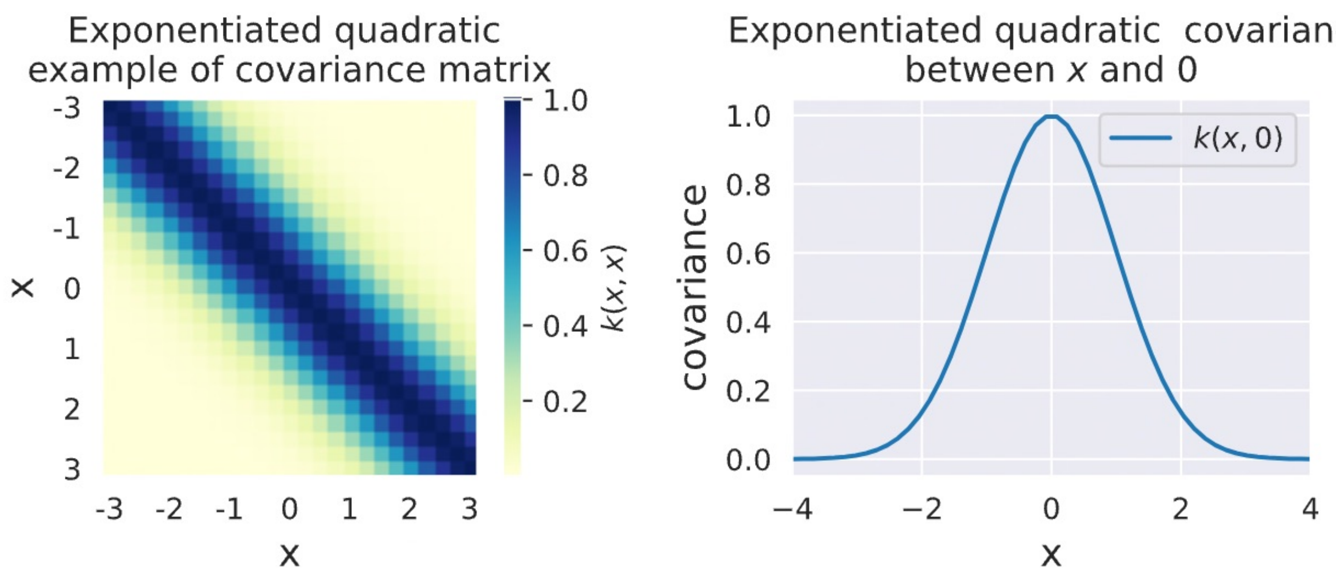
⇒ increase the prob. of y

For $f(x) \sim GP(m(x), k(x,x'))$

covariance or "kernel"

To sample functions with a prior, we specify the kernel

$$K(\underline{x},\underline{x}') = e^{-\frac{1}{2\sigma^2}||\underline{x}-\underline{x}'||^2}$$



<u>In practice</u> how draw sample functions?

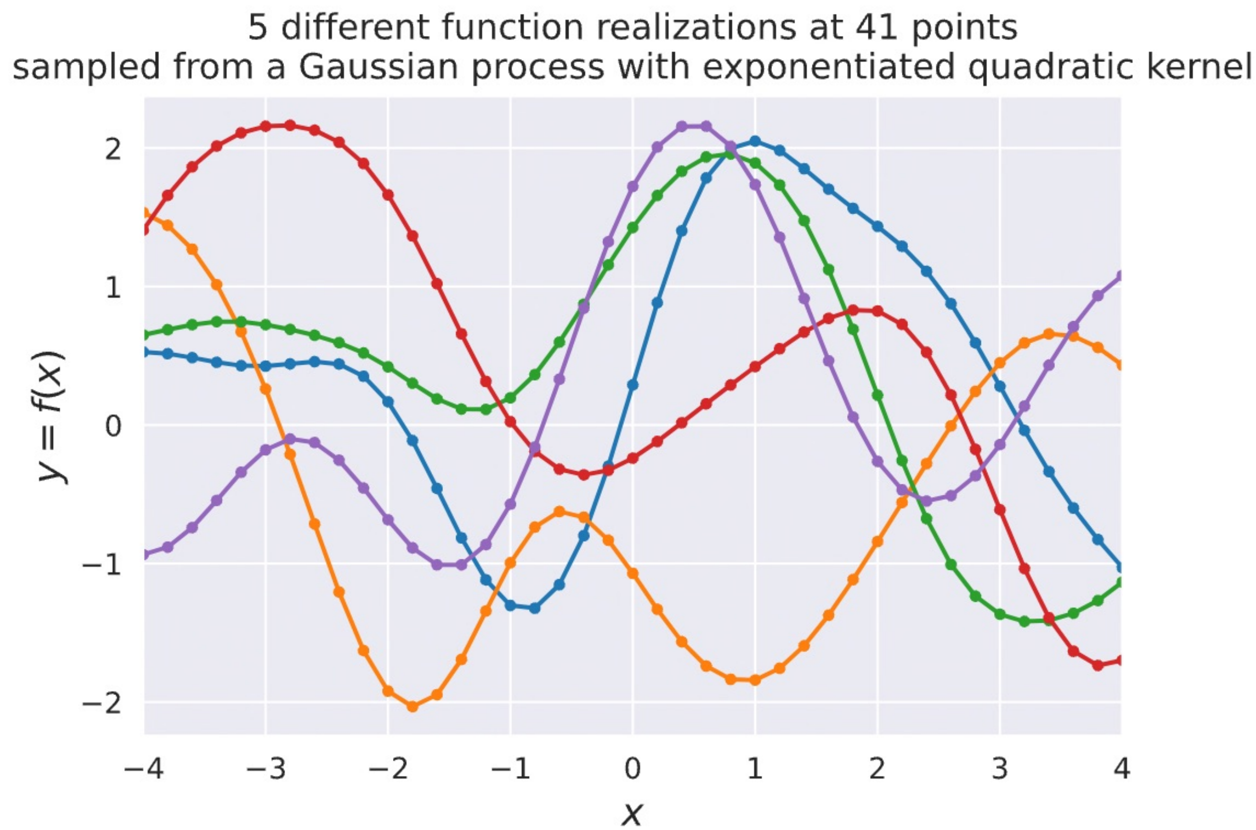
Can't define $f(x)$, would need infinite evaluations of $m(x)$, $k(x, x')$

→ sample at finite $\underline{X}$.

→ gives $\underline{y} = f(\underline{X})$

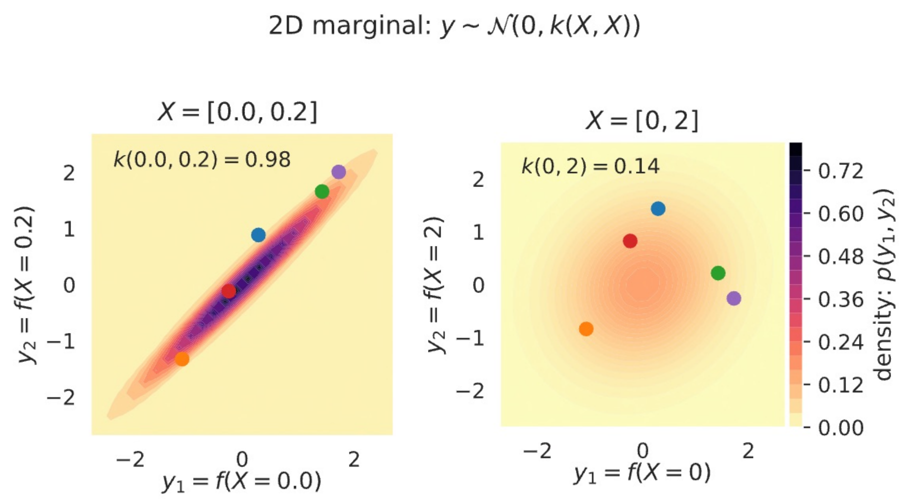
with

$$\underline{y} \sim N(\underbrace{\mu}_{m(\underline{X})}, \underbrace{\Sigma}_{k(\underline{X}, \underline{X})})$$



Take any two of 41:

Points close are highly correlated



Back to Deep Onets

Physics-Informed Deep Neural Operator Networks

Somdatta Goswami[1], Aniruddha Bora[1], Yue Yu[2], and George Em Karniadakis[⋆1,3]

LEARNING THE SOLUTION OPERATOR OF PARAMETRIC PARTIAL DIFFERENTIAL EQUATIONS WITH PHYSICS-INFORMED DEEPONETS

Sifan Wang
Graduate Group in Applied Mathematics
and Computational Science
University of Pennsylvania
Philadelphia, PA 19104
sifanw@sas.upenn.edu

Hanwen Wang
Graduate Group in Applied Mathematics
and Computational Science
University of Pennsylvania
Philadelphia, PA 19104
wangh19@sas.upenn.edu

Paris Perdikaris
Department of Mechanichal Engineering
and Applied Mechanics
University of Pennsylvania
Philadelphia, PA 19104
pgp@seas.upenn.edu

Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators

Lu Lu[1], Pengzhan Jin[2,3], Guofei Pang[2], Zhongqiang Zhang[4] and George Em Karniadakis[2]✉

$$u_D(x) \sim \mathcal{GP}(0, \mathcal{K}((x,y),(x',y'))),$$
$$\mathcal{K}(\boldsymbol{x},\boldsymbol{x}') = \exp[-\frac{(\boldsymbol{x}-\boldsymbol{x}')^2}{2l^2}],\ l = 0.2,\ \text{and}\ \boldsymbol{x},\boldsymbol{x}' \in [0,1].$$

larger $l \longrightarrow$ smoother "u"

Lu Lu
joint work with P. Jin, G. Pang, Z. Zhang, & G. Karniadakis

Division of Applied Mathematics, Brown University

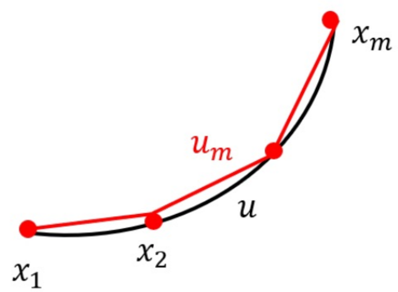
SIAM Conference on Mathematics of Data Science
June, 2020

Convergence w.r.t. the number of sensors

Consider $G : u(x) \mapsto \boldsymbol{s}(x)$ ($x \in [0,1]$) by ODE system

$$\frac{d}{dx}\boldsymbol{s}(x) = \boldsymbol{g}(\boldsymbol{s}(x), u(x), x), \quad \boldsymbol{s}(0) = \boldsymbol{s_0}$$

$$\forall u \in V \Rightarrow u_m \in V_m$$



Let $\kappa(m, V) := \sup_{u \in V} \max_{x \in [0,1]} |u(x) - u_m(x)|$

e.g., Gaussian process with kernel $e^{-\frac{\|x_1 - x_2\|^2}{2l^2}}$: $\kappa(m, V) \sim \frac{1}{m^2 l^2}$

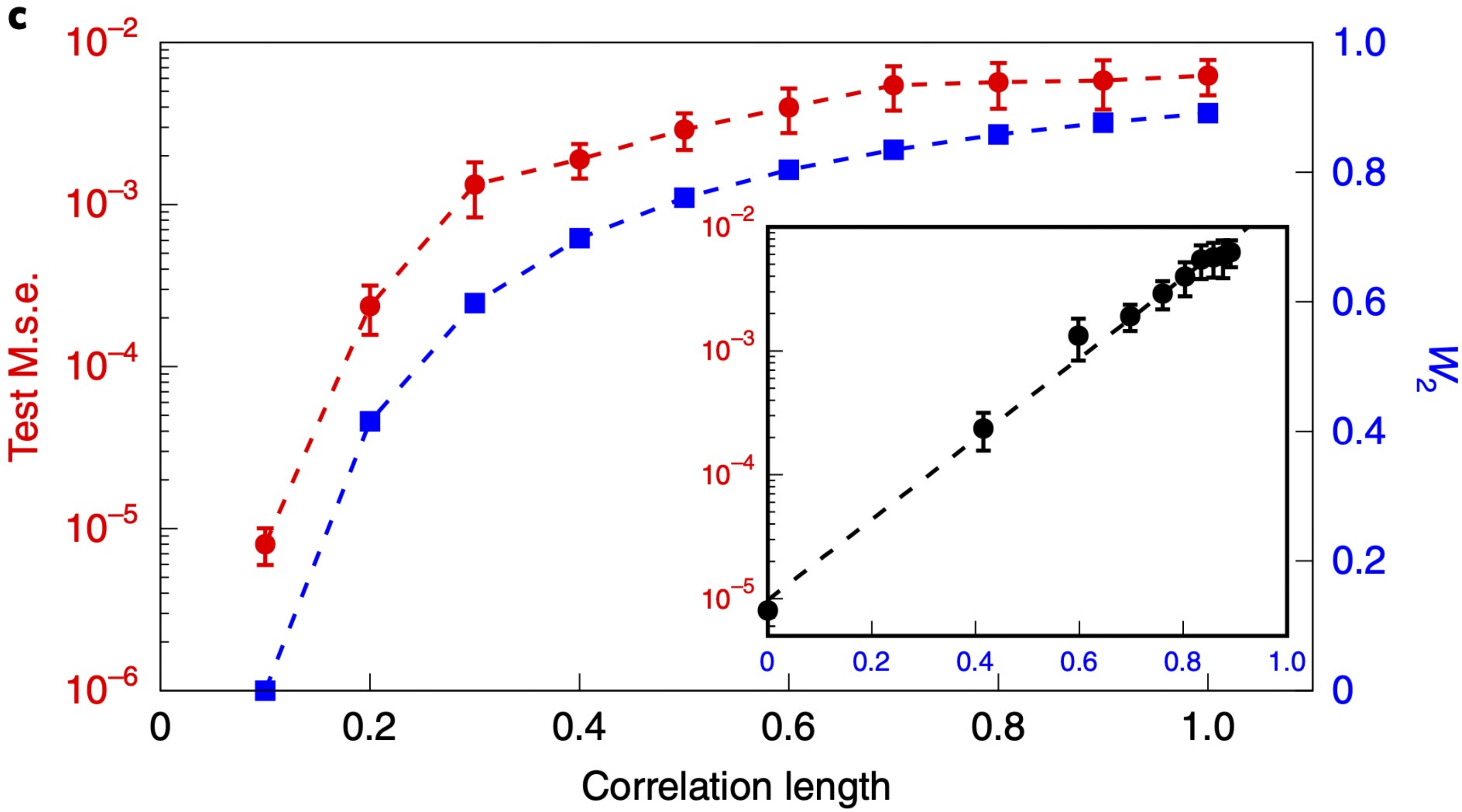
Theorem (Lu et al., 2019; informal)

There exists a constant C, such that for any $y \in [0,1]$,

$$\sup_{u \in V} \|G(u)(y) - NN(u(x_1), \ldots, u(x_m), y)\|_2 < C\kappa(m, V).$$

Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators

Lu Lu[1], Pengzhan Jin[2,3], Guofei Pang[2], Zhongqiang Zhang[4] and George Em Karniadakis[2]



c, The error (mean and standard deviation) tested on the space of Gaussian random fields (GRFs) with the correlation length l = 0.1 for DeepONets trained with GRF spaces of different correlation length l (red curve). The 2-Wasserstein metric between the GRF of l = 0.1 and a GRF of different correlation length l is shown as a blue curve. The test error grows exponentially with respect to the W2 metric (inset).