Computational Statistics

10/27/21

Function Interpolation / Approximation, Numerical Integration

Consider a Gaussian process $f \sim GP(0, k)$, where the kernel k is a "Matérn Kernel":

$$k(x, x') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \frac{r}{\rho} \right)^{\nu} K_\nu \left(\sqrt{2\nu} \frac{r}{\rho} \right)$$

where $r = \|x - x'\|$.

$$\Gamma(\nu) = \text{Gamma function}$$

$$= \int_0^\infty x^{\nu-1} e^{-x} \, dx$$

$$= (\nu - 1)! \quad \text{if } \nu \text{ is positive integer}$$

and $K_\nu(x) =$ modified Bessel function

$$= \frac{\pi}{2} i^{\nu+1} H_\nu^{(1)}(ix)$$

$$= \frac{\pi}{2} i^{\nu+1} \left(J_\nu(ix) + i Y_\nu(ix) \right)$$

and $J_\nu(x)$, $Y_\nu(x)$ are the two solutions to Bessel's equation:

$$x^2 \varphi''(x) + x\varphi'(x) + (x^2 - \nu^2)\varphi(x) = 0 \quad \text{for } x \in [0, \infty)$$

[Do Mathematica demo to plot such functions]

If we actually want to compute with this covariance kernel, we must be able to numerically evaluate it

- accurately
- efficiently

Γ and K_ν can be considered "special functions" - a name given to many functions appearing in classical mathematical physics, etc. Often their numerical evaluation is based on a deep analysis of the governing diff. eq., etc.

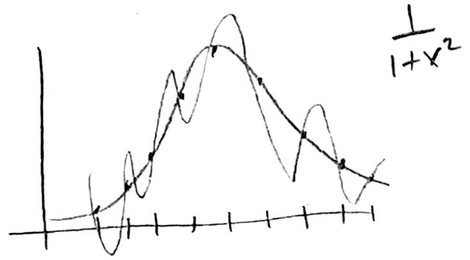
But assuming we can evaluate them, wherever we want, is there a simple way to approximate them and store (i.e. save) the approximation for use at a later date?

Idea 1 Function Interpolation

We all probably know the following: given $n+1$ distinct points $x_0, \ldots, x_n$ and values $y_0, \ldots, y_n$ there exists a unique polynomial of degree n passing through these points:

Interpolation is very sensitive to the location of the nodes $x_0, \dots, x_n$ (assuming the underlying function is smooth)

This is called Runge's Phenomenon [Do Matlab demo]



Short answer: pick better nodes (when possible), i.e. "Chebyshev nodes"

on $[-1,1]$, $x_j = \cos\left(\frac{\pi}{2}\frac{(2j-1)}{n}\right)$, $j = 1, \dots, n$

x_j are the roots of $T_n = \cos(n \operatorname{acos} x)$

These nodes offer the best "minmax" interpolating polynomial:

$$\min_{p_n} \max_{[-1,1]} |p_n(x) - f(x)|$$

Remember: The interpolating polynomial is unique - various ways exist on how to evaluate it

- Lagrange form
- Barycentric Lagrange form.

Idea 2 Function Approximation (similar to "smoothing" in statistics)

Instead of interpolating f, we may want to approximate it as a linear combination of functions $\varphi_1, \varphi_2, \dots$:

$$f(x) \approx \sum_{n=1}^{p} c_n \varphi_n(x)$$

in order to minimize some error, e.g.

$$\underbrace{\| f - \sum_1^p c_n \varphi_n \|_2}_{\text{Least squares problem}}$$

What are the coefficients?

Basic Fourier Analysis

Let $\varphi_1, \varphi_2, \dots$ be an orthonormal basis for $L_2(-1,1)$; i.e., any function $f \in L_2(-1,1)$ can be written as

$$f(x) = \sum_1^{\infty} c_n \varphi_n(x)$$

and

$$\|\varphi_n\|_2 = 1 = \sqrt{\int_{-1}^{1} |\varphi_n(x)|^2 \, dx}$$

$$(\varphi_n, \varphi_m) = \delta_{mn} = \int_{-1}^{1} \varphi_m(x)\, \varphi_n(x)\, dx$$

$\Rightarrow$ coefficients are equal to

$$c_n = (\varphi_n, f) = \int_{-1}^{1} \varphi_n(x)\, f(x)\, dx.$$

One may not always have f though – only samples f_j at points x_j – then least squares can still be done: find $c_1, \dots, c_n$ to minimize $\sum_{j=1}^{n} | f_j - \sum_{n=1}^{p} c_n \varphi_n(x_j)|^2$

(p need not be the same as n...)

What types of functions φ_n are practical?

Fourier : $\varphi_n = \cos nx, \ \sin nx$ on $[0, 2\pi]$

orthogonal polynomials : Legendre, Chebyshev, etc.

For any weight function $w > 0$, and interval $[a,b]$, a set of orthogonal poly's can be constructed such that $\int_a^b p_m(x)\, p_n(x)\, w(x)\, dx = \delta_{mn}$

A special class that comes up in probability / stats: Hermite pols: $w(x) = e^{-x^2}$, like the normal density, on $(-\infty, \infty)$:

$$\int_{-\infty}^{\infty} H_m(x)\, H_n(x)\, e^{-x^2}\, dx = \int_{-\infty}^{\infty} \underbrace{\left(H_m(x)\, e^{-x^2/2}\right)}_{\text{Hermite function}} \left(H_n(x)\, e^{-x^2/2}\right) dx$$

Any function in $L_2(-\infty, \infty)$ can be written as

$$f(x) = \sum_{m=0}^{\infty} c_m H_m(x)\, e^{-x^2/2}$$

Orth. Pols also have many other uses, which we will see (here and in HW).

Back to Matérn:

$$k(r) \approx \frac{1}{\Gamma(\nu)} r^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu}}{\rho} r\right)$$

Γ, K_ν can be approximated / interpolated on the intervals of interest, and saved:

$$K_\nu(r) \approx \sum_{1}^{p} c_n \underbrace{P_n(r)}_{\text{orth. pol.}} \quad \text{on } [0,a] \quad \text{, compute and save coefficients}$$

Note: Interpolation is a form of function approximation, but not all function approximation needs to be an interpolant.

Ex: Interpolation with orthogonal pols at $(x_1, f_1), \dots, (x_n, f_n) \dots$

Enforce that

$$\vec{f}\left\{\begin{aligned} f_1 &= \sum_{j=0}^{n-1} c_j P_j(x_1) \\ &\vdots \\ f_n &= \sum_{j=0}^{n-1} c_j P_j(x_n) \end{aligned}\right\} \begin{pmatrix} P_1(x_1) & \dots & P_n(x_1) \\ & & \\ P_1(x_n) & \dots & P_n(x_n) \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} = P\vec{c}$$

$$\Rightarrow \quad \vec{c} = P^{-1}\vec{f}.$$

The polynomials and location of x_j determine the condition number of P (i.e. the stability of the interpolation).