Explanation of Mathematical Methods Behind the First Half of the Project

AUTHOR
Emma Bowen

Introduction

What mathematical methods will I explain in this document?

- Gaussian processes
- Bayesian History Matching
- Kriging
- Expected Improvement
- Augmented Expected Improvement
- Knowledge Gradient

POSSIBLY WANT TO SHOW THE BITS OF CODE ASSOCIATED WITH EACH METHOD IN THIS DOCUMENT - SHOW IN EACH SECTION, JUST NOTING HERE

Context

We are building on the paper “Using history matching to speed up management strategy evaluation grid searches”. This paper is looking to find the Harvest Control Rule parametrised by F_{target} and $B_{trigger}$ that maximises the catch whilst keeping the risk below 0.05 for a single-stock fishery. The paper does not consider fleet dynamics. We have an objective function in the paper which combines the risk and the catch and so this is the function we want to maximise to get our maximal catch with the constraint of keeping the risk below 0.05.

To do this, the paper takes a Bayesian History Matching (BHM) approach. Firstly, we sample our first eight points which are spaced evenly throughout the sample space to get some initial data. Then, for each round we do the following:

- We set up or update the Gaussian Process (GP) to model the risk and the GP to model the catch
- We can then use the risk as a threshold so that we only consider the values of F_{target} and $B_{trigger}$ that have risk below 0.05
- We use the GP which is modelling the catch to get the value for the catch at every point in the sample space which will have some uncertainty
- We use BHM to remove any points that are implausible (that have a low probability of being higher than the current best catch)

We repeat this process until there is only one plausible point left and then we will accept this as being the F_{target} and $B_{trigger}$ that maximise the catch whilst keeping the risk below 0.05.

Now, we will look at the mathematics behind the methods above and also at the acquisition functions of Expected Improvement, Augmented Expected Improvement and Knowledge Gradient which I added to the original code.

Gaussian Processes

A Gaussian process has a mean function μ_0 and kernel Σ_0 and is a probability distribution over some function g with the property that, for any given collection of points $x_1, \ldots x_m$, the marginal probability distribution on $(g(x_1), \ldots, g(x_m))^T$ is given by:

$$\begin{aligned}(g(x_1), \ldots, g(x_m))^T \sim & N((\mu_0(x_1), .., \mu(x_m)), (\Sigma_0(x_1, x_1), \ldots, \\ & \Sigma_0(x_1, x_m), \ldots, \Sigma_0(x_m, x_1), \ldots, \Sigma_0(x_m, x_m)))\end{aligned}$$

We choose a covariance function such that inputs that have nearby points that have been evaluated have a more certain output than points that are further away from the points that have been evaluated. This is equivalent to saying that if for some x, x', x'' in the design space we have $\|x - x'\| < \|x - x'\|$ for some norm $\| \cdot \|$, then $\Sigma_0(x, x') > \Sigma_0(x, x'')$.

We use a GP to emulate the objective function because it is much cheaper to evaluate compared to our objective function. If we let the emulator be denoted as $\hat{f}$ then we can calculate $\hat{f}(x)$ for any x in the design space as our estimate of $f(x)$ based on our current beliefs. This is true even for the evaluated points $x_1, \ldots, x_n$ as the emulator is fitted to these points. We'll now explore how these GPs can be used to estimate the values of functions in the Kriging section.

Kriging

Kriging is a Bayesian statistical method for modelling functions. We want to focus on the values $f(x_1), \ldots f(x_k)$ where $X := x_1, \ldots, x_k$ is the finite design space. We can represent these values as the vector $f(x_{1:k}) := (f(x_1), \ldots, f(x_k))^T$. This vector is unknown, but we can assume it was drawn at random from a multivariate normal distribution.

We let $\mu_0(x_{1:k})$ be the mean vector which we construct by evaluating μ_0 at each point in X and Σ_0 be the covariance function (equivalently kernel) which we construct by evaluating the covariance function at every pair of points in X. To encode the belief that as the inputs change slightly, the function only changes slightly, we choose Σ_0 such that any points x_i, x_j that are close together in X have a large positive correlation. Then, we can model $(x_{1:k})$ as below:

$$f(x_{1:k}) \sim N(\mu_0(x_{1:k}), \Sigma_0(x_{1:k}, x_{1:k}))$$

where

$$\mu_0(x_{1:k}) = (\mu_0(x_1), \ldots, \mu_0(x_k))^T$$

and

$$\Sigma_0(x_{1:k}, x_{1:k}) = (\Sigma_0(x_1, x_1), \ldots, \Sigma_0(x_1, x_k), \ldots, \Sigma_0(x_k, x_1), \ldots, \Sigma_0(x_k, x_k))^T$$

and N is the normal distribution. Now, if we have evaluated n points such that we have $f(x_{1:n})$ and want to evaluate x_{n+1} we let $k = n+1$ in equation (???). Then, we can compute the conditional distribution of $f(x_{n+1})$ given $f(x_{1:n})$ using Bayes' rule:

$$f(x_{n+1})|f(x_{1:n}) \sim N(\mu_n(x), \sigma_n^2(x))$$

where:

$$\mu_n(x) = \frac{\Sigma_0(x_{n+1}, x_{1:n})(f(x_{1:n}) - \mu_0(x_{1:n}))}{\Sigma_0(x_{1:n}, x_{1:n})} + \mu_0(x_{n+1})$$

$$\sigma_n^2(x) = \Sigma_0(x_{n+1}, x_{n+1}) - \frac{\Sigma_0(x_{n+1}, x_{1:n})\Sigma_0(x_{1:n}, x_{n+1})}{\Sigma_0(x_{1:n}, x_{1:n})}.$$

This conditional distribution $f(x_{n+1})|f(x_{1:n})$ is called the posterior probability distribution for x_{n+1}. We can calculate this for every point in the design space X. This results in a new GP with a mean vector and covariance kernel that depend on the location of the unevaluated points, the locations of the measured points $x_{1:n}$, and their measured values $f(x_{1:n})$. So, we can use this in every round after updating it.

Bayesian History Matching

Once the objective function f has been evaluated at eight points for the first round, Bayesian history matching speeds up the process by removing any points that are implausible. This is done by using Bayes' Theorem to update our beliefs about the value of every unevaluated point based on the values of the points we have evaluated so far. The general process is described below.

Let f be the objective function and x be a point in the sample space. We begin with some uncertainty about $f(x)$. However, we can make probabilistic statements such as:

$$P(f(x) > a) = \int_a^\infty P(f(x))df(x)$$

Once we evaluate another point x' where $ x' x$, we are able to improve our integral to:

$$P(f(x) > a|f(x')) = \int_a^\infty P(f(x)|f(x'))df(x)$$

We now let $a = max\{f(x_1), \ldots, f(x_l)\}$ where l is the number of points we have evaluated so far. For the first round, $l = 8$ but as the rounds increase we make sure to include all previous points of the objective function that have been evaluated. We remove the point x if:

$$P(f(x) > a|f(x_1), \ldots, f(x_l)) = \int_a^\infty P(f(x)|f(x_1), \ldots, f(x_l))df(x) < \varepsilon$$

using $\varepsilon = 0.00001$ until only one plausible point remains as the optimum x^*. In our case, x^* is the F $_{target}$ and B$_{trigger}$ that will give the highest catch whilst following the precautionary principle.

COULD TALK ABOUT HOW TO FIND $P(f(x)|f(x_1), \ldots, f(x_l))$ BUT O/W VERY GOOD

UNSURE IF THIS IS WHAT HAPPENS AND CAN'T SEE SOURCES, but can we replace f with $\hat{f}$ and then say that they have similar enough behaviour by design that we can say that $P(f(x) > a|f(x_1), \ldots, f(x_l)) = P(\hat{f}(x) > a|f(x_1), \ldots, f(x_l))$ as we can easily evaluate $\hat{f}$ because it's our GP and through this we know about f as they are designed to have similar behaviour. Then this also make setting up the emulator makes sense.

Expected Improvement

The first type of acquisition function we will look at is Expected Improvement (EI). At iteration n, we sample the point x_n and observe the value $f(x_n)$. Then, if we were to return a solution at this point, bearing in mind we observe the objective function f without noise and we can only return points we have already evaluated, we would return $f^*_n = max\{f(x_1), \ldots, f(x_n)\}$. If we evaluate at another point x_{n+1} and observe $f(x_{n+1})$ this allows us to define the expected improvement as:

$$EI_n(x_{n+1}) := E_n[f(x_{n+1}) - f^*_n]^+$$

where $[f(x_{n+1}) - f^*_n]^+$ is the positive part of $[f(x_{n+1}) - f^*_n]$ and E_n indicates the expectation taken under the posterior distribution given evaluations of f at $x_1, \ldots, x_n$ so that we are using the previous points we have evaluated to help us find which new point is best to evaluate. This acquisition function is relatively easy to optimise and many different methods have been developed for doing this.

Augmented Expected Improvement

By adjusting Equation (12) found in Global Optimization of Stochastic BlackBox Systems via Sequential Kriging Meta-Models by Huang et al. in 2006 to our own notation, we get that:

$$AEI_n(x_{n+1}) = EI_n(x_{n+1})\left(1 - \frac{\sigma_\varepsilon}{\sqrt{s^2(x_{n+1}) + \sigma^2_\varepsilon}}\right)$$

where σ_ε is the standard deviation of the random error of the objective function f and $s(x)$ is the standard deviation of GP $\hat{f}$ at the n^{th} iteration.

Knowledge Gradient

We remove the assumption of EI that we have to return a pre-evaluated point as our best point. This allows us to do some different computations to the ones in EI. We also now start by saying that the solution we would choose if we have to stop sampling after n points would be the point in the design space with the largest $\mu_n(x)$ value, where μ_n is the mean vector of the posterior probability distribution after n iterations. We call this maximum x^*_n and then can say that $f(x^*_n)$ is random under the posterior distribution and has the conditional expected value:

$$\mu^*_n := \mu_n(x^*_n) = max_x \mu_n(x)$$

where x is any point in the sample space.

Then, we imagine that we are now allowed to sample a new point x_{n+1}. We get a new posterior distribution which we can calculate using equation $(???)$ by replacing x_{n+1} with x and $x_{1:n}$ with $x_{1:n+1}$ to include our new observation. This will have the posterior mean function $\mu_{n+1}(\cdot)$. The conditional expected value for $f(x_n^*)$ changes to be:

$$\mu_{n+1}^* := max_x \mu_{n+1}(x)$$

So, we can see that the increase in the conditional expected value of $f(x_n^*)$ by sampling the new point x_{n+1} is:

$$\mu_{n+1}^* - \mu_n^*$$

While this quantity is unknown before we sample x_{n+1} we can calculate it's expected value given our observations $x_1, \ldots, x_n$. The Knowledge Gradient for sampling at a new point x in the design space is defined as:

$$KG_n(x) := E_n[\mu_{n+1}^* - \mu_n^* | x_{n+1} = x]$$

where again E_n indicates the expectation taken under the posterior distribution at the n^{th} iteration. We would sample the point x with the largest $KG_n(x)$ as our next point.

The easiest way to calculate the KG is via simulation. This can be done by simulating one possible value for $f(x_{n+1})$. Then, we calculate μ_{n+1}^* and subtract μ_n^*. We iterate this process many times so that we can find the average of $\mu_{n+1}^* - \mu_n^*$ and this allows us to estimate $KG_n(x)$. This process, or calculating equation $(???)$ directly from the properties of the normal distribution, both work well in discrete, low dimensional problems which is the situation we are in for the first half of the project.

Alternatively, we can calculate μ_{n+1} using the formula below:

$$\mu_{n+1}(x) = \mu_n(x) + \frac{\mathrm{cov}_n(x_{n+1}, x)}{\mathrm{var}_n(x_{n+1}) + \sigma_{\mathrm{obs}}^2}(F_{n+1} - \mu_n(x_{n+1})).$$

where F_{n+1} is the random distribution for $f(x_{n+1})$ which we tried to approximate above by simulating many times and σ_{obs}^2 is a noise variable which can be determined by the user. From the GP for catch, we get the covariance matrix $\mathrm{cov}_n(x_{n+1}, x)$ and the standard deviation σ_{obs}^2. Then, we have everything needed to compute equation $(???)$.