Currently, we start at some point. Build a local model. Choose a new point. Report the most recent point. AND: See building a local model as an exercise in calculus.

Notice: In this form, might be hard if function evaluations are expensive. We don't have explicit derivatives.
Alternative view:

- exercise in decision making and utility

Start with $D_0 = \{(x_0, y_0)\}$

\[ \text{value at some point} \]

iterate:

- use $D_i$ to predict

\[
\text{value at } x_{i+1}
\]

(a new point to eval at)

form $D_{i+1} = D_i \cup \{(x_{i+1}, y_{i+1})\}$

\[ \text{value at } x_{i+1} \]

Use $D_N$ to predict some point — the result of the optimization

This could be the best point we’ve seen, but could be much more interesting.
Setting all this up requires some machinery. We need to manipulate uncertainty on function values.

Q: I have \( x_i \), what \( x_{it} \) will yield “the best” outcome?

A Gaussian process is a stochastic process, a generalization of the idea of a random vector to functions. A random vector where the mean and covariance of the values of a sample function made by process
is given by

\[
\begin{pmatrix}
m(x_1) \\
m(x_2) \\
\vdots \\
m(x_i)
\end{pmatrix}
\]

for some fixed function \( m \)

\[
\text{cov} = \begin{pmatrix}
K(x_1, x_1) & K(x_1, x_2) & \cdots & K(x_1, x_i) \\
K(x_2, x_1) & K(x_2, x_2) & \cdots & K(x_2, x_i) \\
\vdots & \vdots & \ddots & \vdots \\
K(x_i, x_1) & K(x_i, x_2) & \cdots & K(x_i, x_i)
\end{pmatrix}
\]

for some \( K \).

Notice: \( K \) has to have special properties (at least P.D. for any set of points) giving \( m \), \( K \) yields the process.
Very like a Gaussian for a super-big vector.

A useful trick with Gaussians

\[ p(x_1, x_2) \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}\right) \]

we observe \( x_2 = a \)

Q: What is \( x_1 \mid x_2 = a \) ?

write \( P = \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}^{-1} = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \)

now

\[ \log p(x_1 \mid x_2 = a) = k - \frac{1}{2} \left[ x_1^T P_{11} x_1 - 2 m_1^T P_{11} x_1 \\ + 2 a^T P_{21} x_1 - 2 m_2^T P_{21} x_1 \right] \]

all terms not in \( x_1 \)
Now:
- recall if \( u \sim N(\mu, \Sigma) \)
  then \( \log p(u) = K - \frac{1}{2} (u-\mu)^T \Sigma^{-1} (u-\mu) \)
- match terms:
  \[
  \text{if } p(x_1 | x_2 = a) \text{ is } N(\mu, \Sigma)
  \]
  \[
  \Sigma^{-1} = P_{11}
  \]
  \[
  \mu = m_1 + P_{11}^{-1} P_{12} (a - m_2)
  \]

Now:
\[
\begin{pmatrix}
  P_{11} & P_{12} \\
  P_{21} & P_{22}
\end{pmatrix}
\begin{pmatrix}
  S_{11} & S_{12} \\
  S_{21} & S_{22}
\end{pmatrix}
= I
\]
\[
P_{21} S_{21} = -P_{12} S_{22}
\]
so:
\[
P_{11} S_{11} = I - P_{12} S_{21}
\]
so:
\[
\Sigma = S_{11} - S_{12} S_{22}^{-1} S_{21}
\]
\[
\mu = m_1 + \hat{a} S_{12} S_{22}^{-1} (a - m_2)
\]
All this works for G.P.'s

For example: \( f \) is G.P. \( m, k \)

- know value at \( x_i \) \( \ldots \) \( x_i \) \( (v_i \ldots v_i) \)

Q: What do we know about \( f(x_r) \)

A: it is Normal

\[ f(x_r) \sim N(m_r, \Sigma_r) \]

A: defining property of G.P.'s

\[ \text{and we can write these out explicitly.} \]
\[ y = \begin{pmatrix} v(x_1) \\ \vdots \\ v(x_r) \end{pmatrix} \]

\[ S_{11} = \begin{pmatrix} K(x_1, x_1) \\ \vdots \\ K(x_r, x_r) \end{pmatrix} \]

\[ S_{21} = \begin{pmatrix} K(x_r, x_1) \\ \vdots \\ K(x_r, x_i) \end{pmatrix} \]

\[ S_{12} = \begin{pmatrix} K(x_1, x_r) \\ \vdots \\ K(x_i, x_r) \end{pmatrix} \]

\[ S_{22} = K(x_r, x_r) \]

\[ \Sigma_r = S_{11} - S_{12} S_{22}^{-1} S_{21} \]

\[ \mu_r = m(x_r) - S_{12} S_{22}^{-1} (y - \begin{pmatrix} m(x_1) \\ \vdots \\ m(x_i) \end{pmatrix}) \]

[Plug in and crank!]
a straightforward example:

\[ m = 0 \]
\[ K = (x, x') = e^{-\frac{||x - x'||^2}{2}}. \]

Then

\[ M_r = 0 - \sum_j (v_j) e^{-\frac{||x_r - x_j||^2}{2}} \quad \text{(plug in)} \]

Sample optimization strategy:

given \( x_1, \ldots, x_i \), report the \( x_r \) with largest mean

(\text{so above, with some optimizer to get } x_r)

But where do \( x_1, \ldots, x_i \) come from?

we can get them with a form of dynamic programming!

- First, more detail on representation
The expression for \( f \) shows that:

In GP representation, we can estimate the distribution of function values at a point. Garnett, Fig 2.1, 2.2 below.

**Figure 2.1:** Our example Gaussian process on the domain \( \mathcal{X} = [0, 30] \). We illustrate the marginal belief at every location with its mean and a 95% credible interval and also show three example functions sampled from the process.

**Figure 2.2:** The posterior for our example scenario in Figure 2.1 conditioned on three exact observations.
Facts

1. One can build a joint G.P. for more than one function (constrained or multiobjective opt.)
2. G.P. sample paths (the technical term for the function obtained by tracing a sample from a G.P.) are continuous if \( K(x, x) \) is continuous.

*Diagonal: args are *

3. G.P. sample paths are differentiable if the mean function is diff. AND \( K(x, x) \) is diff.
4. There is a joint dist between function and gradient, assuming sufficiently well-behaved \( m, K \).
5. For compact domain, continuous sample paths, G.P. sample paths have a max.
6. This is unique if no two unique function values are perfectly correlated.
Common m's, K's

\[ m = \text{const.} \]

\[ m = \sum_i a_i \psi_i(x). \]

\[ K(x, x') = \exp \left( -d(x, x') \right) \]

\[ d(x, x') = \sqrt{(x-x')^T(x-x')} \]

\[ K(x, x') = \exp \left( -\frac{d^2}{2} \right) \]

\[ K(x, x') = (1 + \sqrt{3}d) \exp \left( -\sqrt{3}d \right) \]

\[ K(x, x') = (1 + \sqrt{5}d + \frac{5}{3}d^2) \exp \left( -\sqrt{5}d \right) \]

\[ K(x, x') = \sum_i w_i \left[ \exp \left[ -2\pi^2 (x-x')^T \Sigma_i (x-x') \right] \cos \left( 2\pi (x-x')^T \mu_i \right) \right] \]

(\( w_i > 0 \))

There is a very rich theory here for K. Garnett, p 53 to start

The choice is a modelling choice!
Neat trick

Imagine you want \( m(x) = c \), but don't know \( c \).

- put normal prior on \( c \):
  \[ c \sim \mathcal{N}(\mu_c, \Sigma_c) \]

- \( f \) is GP conditioned on \( c \).
- now marginalize out \( c \).

\[ f | c \sim \text{GP}(c, K) \]

so

\[ f \sim \text{GP}(a, K + \Sigma_c) \]

this works for linear combination of basis functions, too (Garnett, p48)
We can now see optimization as subject to decision theory.

Alg:
- start w/ D. (which could be empty)
  repeat:
    \[ x = \text{policy}(D) \] (choose where to observe)
    \[ y = \text{observe}(x) \] (observe)
    \[ D = D \cup \{(x,y)\} \]
  until some termination condition
  return D
  report \[ \text{predict}(D) \].

Q:\[
\begin{align*}
predict? & | \quad \text{Many choices} \\
policy? & | \quad \\
\end{align*}
\]
Start with simplest
predict \((D) = \left[\text{the \(x\) w/ largest function value in \(D\)}\right]\)

Now assume we have \(D_{i-1} = \{(x_i, y_i), \ldots\}\)

And we get to add one point
to get \(D_i\) — which point should we add?

Point to
utility of \(D_{i-1} = \max (y_i, \ldots, y_{i-1})\)

Cans that's what we'd report

We don't know function value at any other point, but we can compute expectations over function values
if we pick $x$, then

\[
\text{utility of } D_i = \text{utility of } D_{i-1} + \max(y - \hat{y}_{i-1}, 0)
\]

the function value at $x$  
the best value in $D_{i-1}$

we can't compute this, but we can compute

\[
\mathbb{E}_{y|x} \left[ \max(y - \hat{y}_{i-1}, 0) \right]
\]

(expected value of the marginal utility of choosing $x$)

what you add to utility of $D_{i-1}$
Picture:

i) if you had only D₃, would report x₃

ii) what x to add?

iii) zoom:

3 sd bars contain x₁, x₂, x₃

mean

x₁, x₂, x₃

contributes to utility of x

zero to utility
Computing the expected marginal utility

We have $\hat{y_{i-1}}$

Want $\frac{1}{\sqrt{2\pi} \sigma} \int_{-\infty}^{\infty} \max(y - \hat{y}_{i-1}, 0) e^{-\frac{(y-m)^2}{2\sigma^2}} dy$

Where $\sigma, m$ are functions of $x$.

This integral is nasty, but can be evaluated

Transform $z = \frac{y - m}{\sigma}$

to get

$\frac{1}{\sqrt{2\pi}} \int_{\frac{\hat{y}_{i-1} - m}{\sigma}}^{\infty} [\sigma z - (m + \hat{y}_{i-1})] e^{-z^2/2} dz$
With some work, we can write in terms of error functions, etc.

\[ g(u) = E_{y_i|x_i=u} \left[ \max(y_i - \hat{y}_{i-1}, 0) \right] \]

- to find the best \( x_i \), max this wrt \( x \)

- Now consider \( D_{i-2} \).
  - we want to add two points
    - one to get to \( D_{i-1} \), then
    - one to get to \( D_i \)

We can find a point \( x_{i+1} \) by assuming we do the best thing with \( D_{i-1} \)
Notice, for any particular $x_{i-1}$ we could

1. eval $y_{i-1}$
2. from that, choose $x_i = \arg\max_u g(u \mid y_{i-1})$
3. but we don't want to eval $y_{i-1}$
4. we have $P(y_{i-1} \mid x_{i-1}, D_{i-2})$

notice

$g(u \mid y_{i-1}, x_{i-1}, D_{i-2})$

$q_i(u)$ depends on:

1. $y_{i-1}$ (because $y_{i-1}$ might change)
2. $x_{i-1}$ (because this affects $\sigma, m$)

write:

$g_i(u \mid y_{i-1}, x_{i-1}, D_{i-2})$
and notice that, once we've chosen $x_{i-1}$, we would choose the best $x_i$.

So we could evaluate $x_{i-1}$ by

$$E_{y_{i-1} \mid x_{i-1}} \left[ \max_u g(u \mid y_{i-1}, x_{i-1}, D_{i-2}) \right]$$

(assuming the expected value of $x_{i-1}$, you then choose the best $x_i$.

An induction follows **but** all should look unpromising

$$E \left[ \max \left( E \left[ \max \ldots \right] \right) \right]$$

should look very hard to evaluate
fairly practical strategy

- One point lookahead.
- go from $D_i$ to $D_{i-1}$ by
  - choosing $x_i = \arg\max_u g(u; D_{i-2})$
  (so ignore what you do with point in future)
  - then $x_i = \arg\max_u g_i(u; D_{i-1})$
- Clearly, you could start at $D_0$
- This is a fair strategy
  - notice how it will cause $x_i$'s to probe places where the function has a strong chance of being better than anything we've seen.
There is a very rich family of methods.

- change procedure used to make final recommendation from i
  - change predictive procedure used to select next x.
  - incorporate noisy function evaluations
  - incorporate derivatives
  - etc

Examples follow:

Procedure used to make final rec.

- have $D_i \rightarrow$ predict what?
- we have seen predict (largest observed
  - $y$ (simple reward))
- alternative predict $x$ s.t. $\mu(x|D_i)$
  - is largest
    - (global reward)
we have an expression for
\[ f(x \mid D_i) \sim N(\mu_r, \Sigma_r) \]
(on p7, p8)

so: choose x w/largest \( \mu_r \)

how: conventional optimization
(eg Newton, etc)

Alternative: imagine we are risk-averse

we would like an \( x \) w/ large \( y \),
But dislike too variable a reward
we might
choose \( x \) w/ largest
\[ \mu_r + \beta \sigma_r \]

(if \( \beta \) -ve, we are avoiding risk,)
(\( \beta \) +ve, " " seeking ")

choosing \( \beta \) is a modelling option
Change decision procedure - rollout

- Assume we have $D_{i-2}$ and want to predict $x_{i-1}$, $x_i$ to get $D_i$.
  - We saw optimal policy (hard)
  - Easier policy (not optimal)

- We could evaluate the suitability of some $x$ to be $x_{i-1}$ by:

  - Draw sample from predictive dist for $f(x|D_{i-2})$ to get $y_{i-1}$.
  - Use this and easy policy to predict $x_i$.
  - Draw sample to get $y_i$.
  - Evaluate resulting $D_i$.

Repet this, and average.

This is rollout.
how then to choose $x_i$?
rollout estimates at a bunch of sample points
fit some kind of approximate function
choose max

One option here is a G.P.! but we have noisy estimates of the function values?
G.P. with noisy observations:

- two applications
  - model rollout values
  - perhaps our function values are noisy?

Recall procedure to obtain predictive dist at \( x \), conditioned on \( D_i \):

\[
y \sim N\left( \begin{pmatrix} m(x) \\ m(x_i) \\ \vdots \end{pmatrix}, \begin{pmatrix} K(x, x) & K(x, x_i) \\ K(x_i, x) & K(x_i, x_i) \end{pmatrix} \right)
\]

\[
N\left( \begin{bmatrix} \mu_r \\ \mu_i \end{bmatrix}, \begin{bmatrix} K_{rr} & K_{ri} \\ K_{ir} & K_{ii} \end{bmatrix} \right)
\]

Now condition on \( \alpha = \gamma \) and use pt. pt.
We now must condition on

\[ 0 = \mathbf{v} + \xi \]

\[ \text{Noise} - N(0, \Sigma_n) \]

notice noise isn't correlated to g.p.

so

\[ y \sim N \left( \begin{bmatrix} \mu_r \\ \mu_i \end{bmatrix}, \begin{bmatrix} K_r & K_{ri} \\ K_{ir} & K_{ii} + \Sigma_n \end{bmatrix} \right) \]

and we use RF, PG !