SummaryPresentation

September 20, 2016

In [2]:
```
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import pandas as pd
# import sys
# sys.path.append("BayesianOptimization/")
from bayes_opt import BayesianOptimization
from matplotlib import gridspec
sns.set_context('talk')
sns.set_style('whitegrid')
%matplotlib inline

def posterior(bo, xmin=-2, xmax=10):
    xmin, xmax = -2, 10
    bo.gp.fit(bo.X, bo.Y)
    mu, sigma2 = bo.gp.predict(np.linspace(xmin, xmax, 1000).reshape(-1, 1)
    return mu, np.sqrt(sigma2)

def plot_gp(bo, x, y):

    fig = plt.figure(figsize=(16,8))
    fig.suptitle('Gaussian Process and Utility Function After {} Steps'.for

    gs = gridspec.GridSpec(2, 1, height_ratios=[3, 1])
    axis = plt.subplot(gs[0])
    acq = plt.subplot(gs[1])

    mu, sigma = posterior(bo)
    axis.plot(x, y, linewidth=3, label='Target')
    axis.plot(bo.X.flatten(), bo.Y, 'D', markersize=8, label=u'Observations
    axis.plot(x, mu, '--', color='k', label='Prediction')

    axis.fill(np.concatenate([x, x[::-1]]),
              np.concatenate([mu - 1.9600 * sigma, (mu + 1.9600 * sigma)[::
        alpha=.6, fc='c', ec='None', label='95% confidence interval')

    axis.set_xlim((-2, 10))
```

```
    axis.set_ylim((0., 1.5))
    axis.set_ylabel('f(x)', fontdict={'size':20})
    axis.set_xlabel('x', fontdict={'size':20})

    utility = bo.util.utility(x.reshape((-1, 1)), bo.gp, 0)
    acq.plot(x, utility, label='Utility Function', color='purple')
    acq.plot(x[np.argmax(utility)], np.max(utility), 'd', markersize=15,
             label=u'Next Best Guess', markerfacecolor='gold',
             markeredgecolor='k', markeredgewidth=1, clip_on=False)
    acq.set_xlim((-2, 10))
    acq.set_ylim((0, np.max(utility) + 0.5))
    acq.set_ylabel('Utility', fontdict={'size':20})
    acq.set_xlabel('x', fontdict={'size':20})

    axis.legend(loc=2, bbox_to_anchor=(1.01, 1), borderaxespad=0.)
    acq.legend(loc=2, bbox_to_anchor=(1.01, 1), borderaxespad=0.)
```

```
In [29]: from traitlets.config.manager import BaseJSONConfigManager
         path = "E:/ProgramFiles_w/Engineering/Anaconda/etc/jupyter/nbconfig"
         cm = BaseJSONConfigManager(config_dir=path)
         # # cm.update('livereveal', {
         # #          'width': 960,
         # #          'height': 700
         # # })
         #                'theme': 'sky',
         #                'transition': 'zoom',
         cm.update('livereveal', {'start_slideshow_at': 'selected'})

         # from IPython.core.display import display, HTML
         # display(HTML("<style>.container { width:80% !important; }</style>"))
```

```
Out[29]: {'start_slideshow_at': 'selected'}
```

1 Thurston Sexton

Sandia NL Interview
9/27/2016

- Optimization
- Numerical Modeling
- Machine Learning

1.1 Learning Human Search Strategies

1.1.1 From a Crowdsourcing Game

Thurston Sexton, Max Yi Ren

Design Informatics Lab
Dept. of Mechanical and Aerospace Engineering
Arizona State University
Presented at IDETC 2016 (Charlotte, Aug.22)
Submitted to ASME JMD *(under review)*

1.1.2 Overview

- **Why *human* search strategy?**
- Bayesian Optimization (IBO)
- Inverse Bayesian Optimization (IBO)
- Case Study: Learning from others vs by one's-self
- IBO vs. Inverse Reinforcement Learning

1.1.3 Why *human* search strategy?

Human beings can **sometimes outperform** optimization algorithms at searching

- e.g. Figuring out how to play a game well.
  - Machine achieved *lower* than human level performance after 38 days of continuous play of ATARI Frostbite (Mnih et al.2015)
  - State-of-the-art deep learning achieves 3.5% of human performance on this game after the same amount of game play (2 hours)

Lake et al., "Building Machines That Learn and Think Like People", arXiv (July 2016)
ecoracer.herokuapp.com

1.1.4 Why do humans succeed where algorithms fail?

- **Knowledge** - Prior information (or physical intuition)?
- **Meta Game** - ability in absraction?
- **Learning to Learn** - Self improvement/strategic insight?

1.1.5 What about Crowdsourcing?

- **Currently**: wait for *good solutions* to arise by watching many players attempt solutions.
  - *a.k.a. smart people spending lots of time on a game*
- **What if**: wait for *good search strategies* by observing fast-improving players' decisions.
  - *a.k.a. smart people willing to at least try the game*

→ **Problem:** If we assume Humans search using an optimization scheme (e.g. Bayesian Optimization) with parameters informed by prior knowledge (their "strategy"), can we recover those parameters using their search trajectory?

1.1.6 Overview

- Why *human* search strategy?
- **Bayesian Optimization (BO)**
- Inverse Bayesian Optimization (IBO)
- Case Study: Learning from others vs by one's-self
- IBO vs. Inverse Reinforcement Learning

1.1.7 What is Bayesian Optimization?

Efficient Global Optimization (EGO) is excellent at optimizing expensive functions with minimal calls. 1. Sample the solution space 2. estimate/update the GP (kriging model) 3. Find new sample by maximizing a utility/aquizition function 4. repeat 2,3

Utility function can be propability of improvement, **expected improvement (EI)**, upper confidence bound, etc.

Search is therefore governed by GP parameters (λ) and the probabilistic nature of optimizing EI (strict vs non $\to \alpha$).

1.1.8 What is BO? Simple 1-D Example

$$\min_x f(x) \quad \text{where} \quad f(x) = e^{-(x-2)^2} + \frac{1}{10}e^{-(x-6)^2} + \frac{1}{x^2+1}$$

```
In [16]: # Our Objective Function
         def target(x):
             return np.exp(-(x - 2)**2) + np.exp(-(x - 6)**2/10) + 1/ (x**2 + 1)

         x = np.linspace(-2, 10, 1000)
         y = target(x)
         bo = BayesianOptimization(target, {'x': (-2, 10)})

         # Gaussian Process Parameters
         gp_params = {"corr": "cubic", "theta0": 0.1, "thetaL": None, "thetaU": Nor
         # Give the Model 3 random points to initialize
         bo.maximize(init_points=3, n_iter=0, acq='ei', xi=0.05, **gp_params)
```

```
Initialization
-----------------------------------------
 Step |   Time |      Value |         x |
    1 | 00m00s |    1.01744 |    5.6372 |
    2 | 00m00s |    1.31674 |    1.6754 |
    3 | 00m00s |    1.26393 |    2.4163 |
Bayesian Optimization
-----------------------------------------
 Step |   Time |      Value |         x |
```

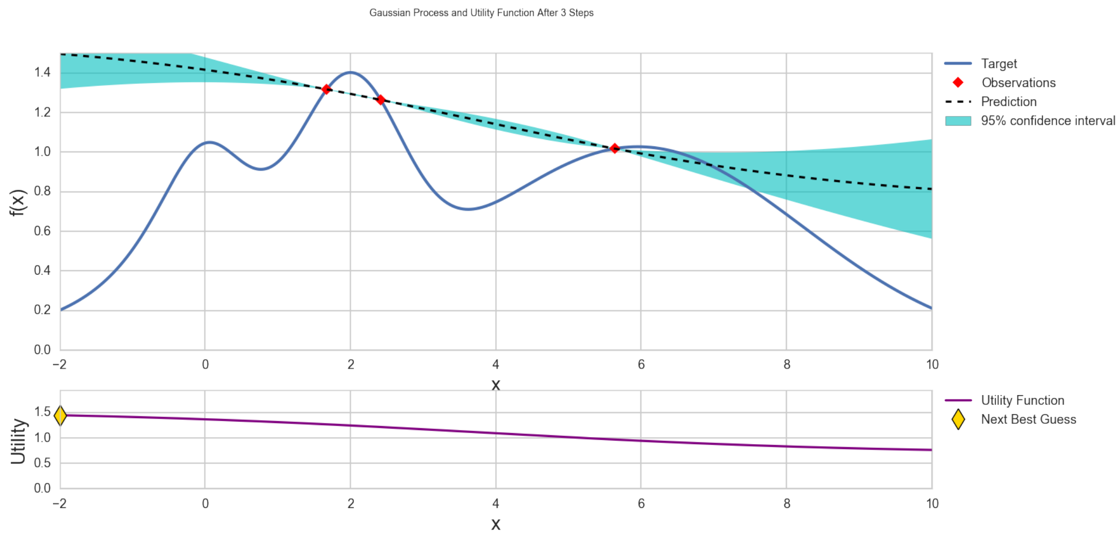
For the set of available samples $\mathbf{X}_k$ and associated objective values $\mathbf{f}_k$,

$$h_k :=< \mathbf{X}_k, \mathbf{f}_k >$$

Then given a utility function Q_{EI},

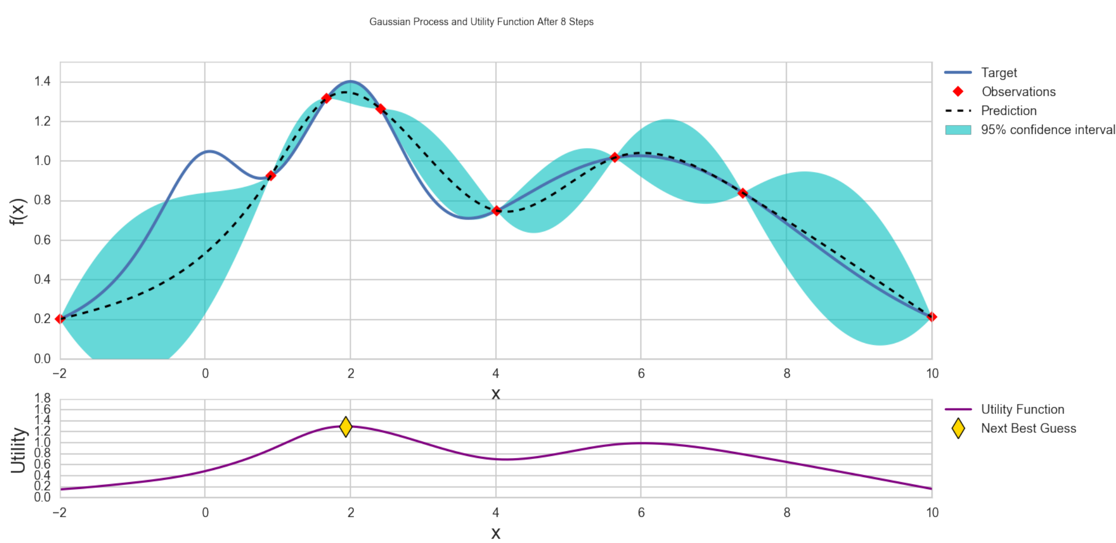
$$x_{k+1} = \text{argmax}_{x \in \mathcal{X}} Q_{EI}(x; h_k, \lambda)$$

In [17]:
```
plot_gp(bo, x, y)
```



In [22]:
```
#Run until convergence
bo.maximize(init_points=0, n_iter=1, xi=.05)  # random init, pts to add, p
plot_gp(bo, x, y)
```

```
Bayesian Optimization
-----------------------------------------
 Step |   Time |      Value |         x |
    8 | 00m00s |    0.83950 |    7.4019 |
```



1.1.9 What is BO? Pros/Cons

- fewer objective function calls
- creates *"surrogate model"* with cheap evaluation
- estimate of certainty is included
- matches actual human search in 1-D (*Borji & Itti 2013*)

but… - needs good Gaussian Process parameters for efficiency - curse of dimensionality
we'll be coming back to these

1.1.10 Overview

- Why *human* search strategy?
- Bayesian Optimization (BO)
- **Inverse Bayesian Optimization (IBO)**
- EcoRacer - Crowdsourced Strategy

We want to recover λ (and α) from an observed trajectory $\mathbf{h}$
Assume it's static. The Log-likelihood of a GP parameter can be defined as:

$$L(\lambda, \alpha) = \sum l(\alpha, \lambda, \mathbf{X}, \mathbf{f})$$

where l is the log-probability of every new sample *given* all previous samples/responses. **Every calculation of l needs *integration* over x**

1.1.11 IBO: Does it work?

Rosenbrock 30-D with four settings…how many samples do we need?

IBO quickly recognizes the true BO settings **unless** BO is behaving ~ random sampling (high λ)

1.1.12 IBO: Is it efficient?

Generally solvers find a λ at each iteration (adaptive) without prior knowledge (Maximum Likelihood Estimate)

1.1.13 Overview

- Why *human* search strategy?
- Bayesian Optimiztion
- Inverse Bayesian Optimization (IBO)
- **EcoRacer - Crowdsourced Strategy**
  - Dimension reduc
  - Results
  - IBO vs. Inverse Reinforcement Learning

1.1.14 EcoRacer

Part design (gear ratio), part control (trinary signal) Treated as a design optimization problem means **very high** dimensionality. - 18160 "ticks" where a descision can be made, spread over 36s - Nearly impossible to do BO without dimension reduction. - Previously: inputs like hill gradient, time since beginning, velocity, etc.

How do Humans "reduce the dimensionality" of a problem? *Salient features*, with minimal amount of multitasking.

→ **Independent Component Analysis (ICA)**

Unlike PCA (maximize variance in orthogonal bases), ICA performs the task of *blind source separation*. For an assumed number of *source signals* ICA finds sources such that they combine to produce the observed signals with **minimum shared information** (K-L divergence).

```
In [ ]:
```

1.2 Numerical Modeling & Simulation

1.2.1 Solving PDE's

Ferromagnetic Phase Separation

MAE 598 (Spring 2016, Dr Yang Jiao)

CFD spinny thingies

MAE 561 (Fall 2015, Dr. Marcus Hermann)

```
In [ ]:
```