PRML本の６章に詳細あり。式番号は、PRML本に合わせている。

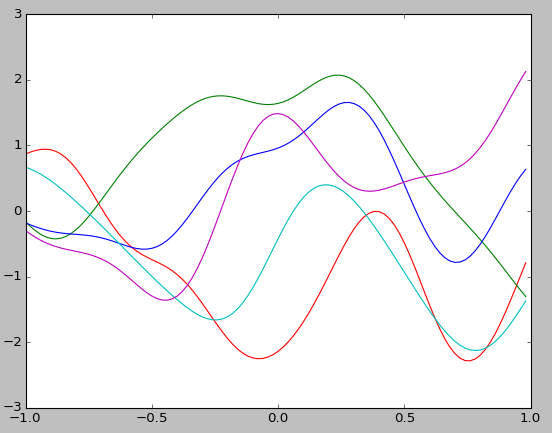
# Gaussian process

とりあえず難しい数式の話は後回しね。データポイントに対して、真の値をとすると、各値は以下の確率で与えられるとする。

ただし、

※なんで、平均が0なの？ってところがすごい混乱するんだけど、そういうものと考えるしかないみたい。逆に言うと、元の値が偏っていたりする場合は、平均が0になるよう調整する必要がある。

下図に、いくつかの曲線が表示されている。１つの曲線が、１つのに相当する。ガウス分布に従った乱数なので、平均0から少し離れたりしているわけだよね。



ガウス過程を使ったregressionでは、与えられた観測値に対して、最も近い曲線を当てはめるというイメージだ。※ハッキリ言って、Gaussian processそのものは良く分からんけど、次章のregressionさえ理解しておけば問題ないと思う。

# Gaussian process for regression

データポイントに対して、真の値を、観測値とする。観測値は、真の値に対してノイズが加味されると考える。つまり、

(6.57)

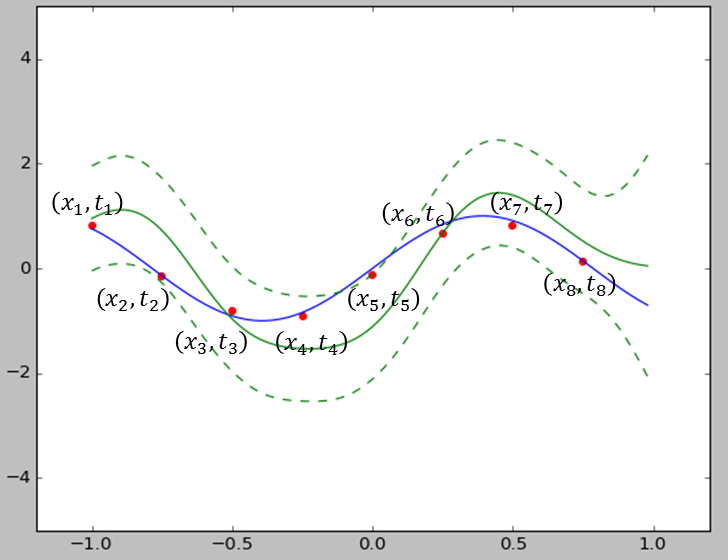
ガウスノイズと仮定し、分散とすると、

(6.58)

また、データポイントの共分散を以下のように定義する。

(6.63)

なんでこんな式を共分散として使用するのか？よく使われる式みたいなので、とりあえず使っとけって感じ。難しい数式はここまで。後は実際にどう使うかの話だ。



この時、任意の点の値は、次のようにして推定される。

(6.66)

ただし、

また、

※式(6.66)は、こう考えると理解できる。点の値は、既に観測したデータ群の加重平均的な感じで計算され、その重みは、点に近い点の観測値をより大きい重みにする。すごく自然な考え方だよね。

推定された値の精度は、以下のように偏差を計算して推定できる。

(6.67)

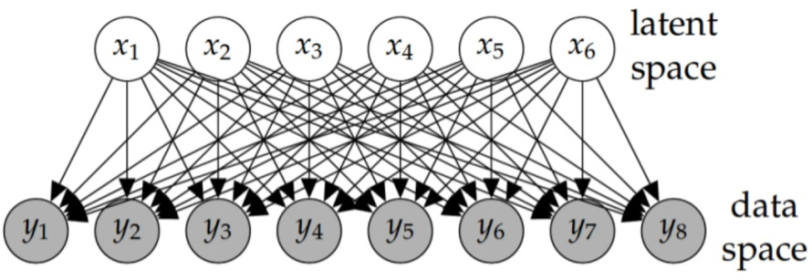
ただし、

とりあえず、以上の式を使うことで、regressionは行えるのだ。とりあえず、「使う」分にはこれで困らないよね。

# Gaussian Process Latent Variable Model (GP-LVM)

そもそも、Latent variable modelって、Machine Learningの授業でもやったけど、なんらかの隠れ変数を想定し、実際に観測される値は、その隠れ変数のマッピングだという考え方だよね。

Graphical modelで表現すると、以下のイメージだ。



では、どうやって隠れ変数を抽出するのか？１つの方法はPCAだ。しかし、PCAは線形変換にしか対応しない。これに対して、GP-LVMは線形である必要がない。

MatlabやC++ライブラリが、以下のページにあるみたい。著者のNeil Lawrenceは、GP-LVMの専門家みたいで、「Learning a Manifold of Fonts」の論文でも引用されている。  
http://ml.sheffield.ac.uk/~neil/gplvmcpp/

欠点としては、non-convex関数に対するパラメータ最適化を行うため、必ずしもglobal optimumを得られるとは限らない。また、大規模データに対して、計算コストが大きくなり、実用的でなくなる。