

Stat 471: Lecture 34

Gaussian Processes II.

Moo K. Chung mchung@stat.wisc.edu

December 5, 2003

1. *White noise* is defined as a random process whose covariance function is proportional to the Dirac-delta function δ , i.e. $R(t, s) \propto \delta(t - s)$. We can define it as the limiting density function of $N(0, \sigma^2 I_N)$ as $\sigma \rightarrow 0$. Sometimes this is taken as the definition of Dirac-delta. *White Gaussian noise* is a white noise whose linear filtering is a Gaussian process. This is the usual definition of white Gaussian noise that can be used in simulating Gaussian processes. In practice, we use the discrete white Gaussian noise which is simply $N(0, \sigma^2 I_N)$ random variables. The discrete version of kernel smoothing is

$$\epsilon(t) = \sum_{i=1}^m K(t - \tau_i) w(\tau_i)$$

where τ_i are regular grid points in $\mathbb{R}^N$. Assume further that $w(\tau_i) \sim N(0, \sigma^2 I_N)$ and $w(\tau) = 0$ if $\tau \neq \tau_i$. Since any linear combination of Gaussian is again Gaussian, $\epsilon(t_1), \dots, \epsilon(t_m)$ must be multivariate normal. Therefore, process $\epsilon(t)$ constructed in this way must be a Gaussian process.

```
kernel=inline('exp(-(x.^2+y.^2)/
               (2*sigma^2))')
dx=kron(ones(5,1), [-2:2]);
dy=kron(ones(1,5), [-2:2]');
weight=kernel(0.8,dx,dy)/
        sum(sum(kernel(0.8,dx,dy)))
smooth_e=conv2(e, weight, 'same');
y2=g+0.2*smooth_e;
```

2. The above example shows how to simulate a Gaussian process in $\mathbb{R}^N$. What if you are supposed to simulate a Gaussian process in a set $M \in \mathbb{R}^N$. Suppose your observation is the coordinates of n points that form a curve. In our example this is the corpus callosum of the human brain. Let $p_i = (x_i, y_i)$ be the coordinates. We can parameterize the curve by t . Then we can have the following model

$$X(t) = \mu_x(t) + \epsilon_x(t),$$

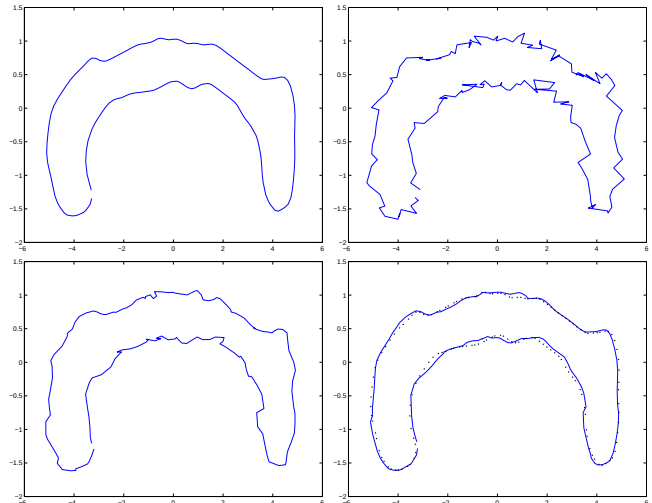
$$Y(t) = \mu_y(t) + \epsilon_y(t).$$

where observations are taken at each t_i so that

$$x(t_i) = x_i, y(t_i) = y_i.$$

If you simply assign iid Gaussian noise to the coordinates $\epsilon_x(t_i) \sim N(0, \sigma_1^2)$ and $\epsilon_y(t_i) \sim N(0, \sigma_2^2)$, the curve does not look very realistic.

```
load CCcurve;
x=CCcurve(:,1);
```



```
y=CCcurve(:,2);
ex=0.2*normrnd(0,1,154,1);
ey=0.1*normrnd(0,1,154,1);
plot(x+ex,y+ey)
```

So we apply Gaussian kernel smoothing to the iid noises and generate Gaussian processes or correlated noises

$$\epsilon_x(t) = K * w_x(t), \epsilon_y(t) = K * w_y(t)$$

where $w_x(t_i) \sim N(0, \sigma_1^2)$ and $w_y(t_i) \sim N(0, \sigma_2^2)$.

```
dx=x(2:154)-x(1:153); dx=[ x(1)-x(154) ;dx];
dy=y(2:154)-y(1:153); dy=[ y(1)-y(154) ;dy];
distance = sqrt(dx.^2 + dy.^2)
for j=1:5
    for i=2:153
        nbr_dist=[distance(i-1) 0 distance(i)];
        weight= kernel(nbr_dist,0.2);
        smooth_ex(i) = dot(ex(i-1:i+1),weight);
        smooth_ey(i) = dot(ey(i-1:i+1),weight);
    end;

    nbr_dist=[distance(153) 0 distance(154)];
    weight= kernel(nbr_dist,1);
    nbr_x= [ex(153) ex(154) ex(1)];
    nbr_y= [ey(153) ey(154) ey(1)];
    smooth_ex(154) = dot(nbr_x,weight);
    smooth_ey(154) = dot(nbr_y,weight);
    % you put similar code for end point p1
    ex=smooth_ex;
    ey=smooth_ey;
end;
```