Gaussian Process: band-limited stochastic process, Shannon reconstruction theorem (statistical version)

Objective: illustrating the "statistical" (Gaussian Process Regression) interpretation of Shannon sampling theorem and associated "sinc" reconstruction with a band-limited Gaussian stochastic process.

*Executed in Matlab `R2024a`

Presentations in Video:

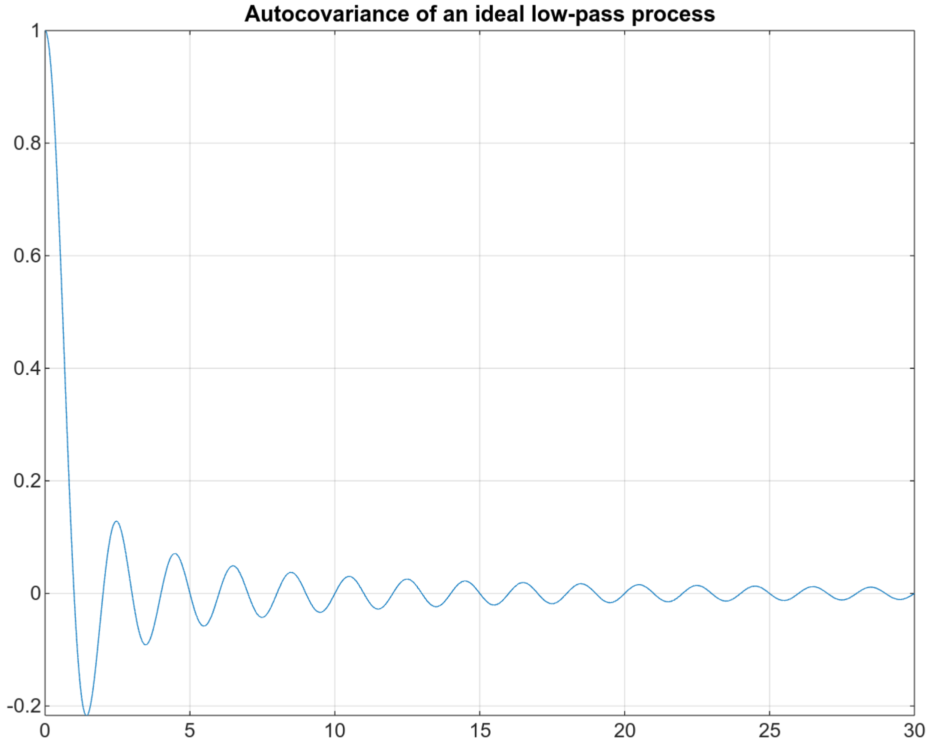
https://personales.upv.es/asala/YT/V/shannonGPEN.html

https://personales.upv.es/asala/YT/V/shannonGP2EN.html

Table of Contents

Covariance kernel and PSD

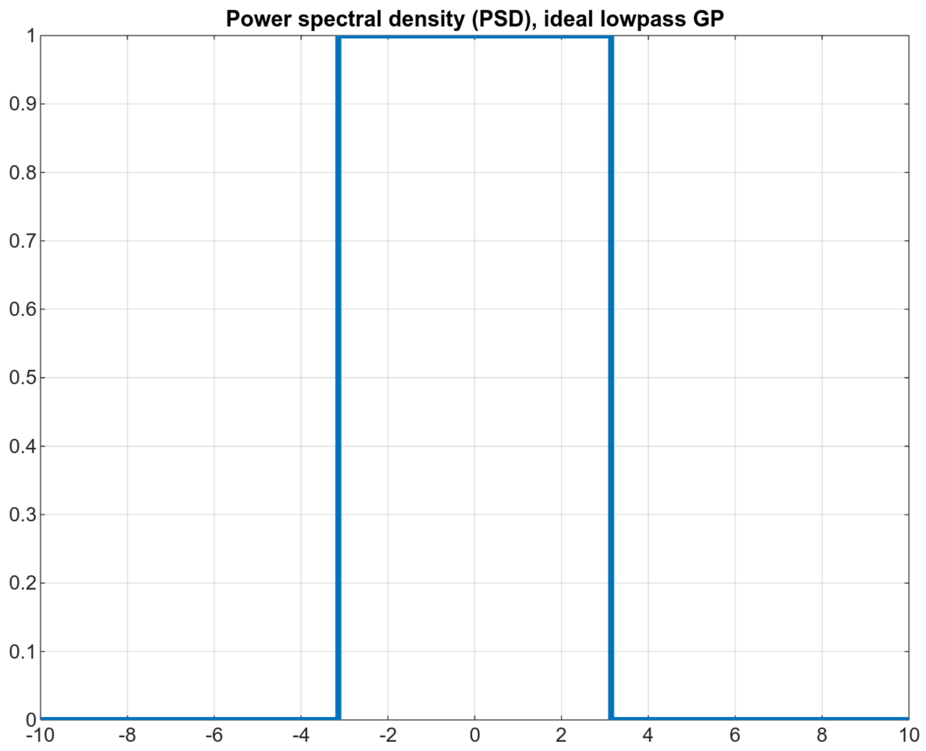
```
kappa=@(d) (sinc(d));
xTest=(-15:0.05:15)';
fplot({kappa},[0 max(xTest)-min(xTest)]), grid on
title("Autocovariance of an ideal low-pass process")
```




```
syms x real
psd1=simplify(fourier(kappa(x)))
```

psd1 = $\mathrm{heaviside}(w+\pi) - \mathrm{heaviside}(w-\pi)$

```
fplot({psd1},[-10 10],LineWidth=3);grid on
title("Power spectral density (PSD), ideal lowpass GP")
```



Example realizations

At test points:

```
Kytest=K(xTest,xTest,kappa)+9e-15*eye(length(xTest));
size(Kytest)
```

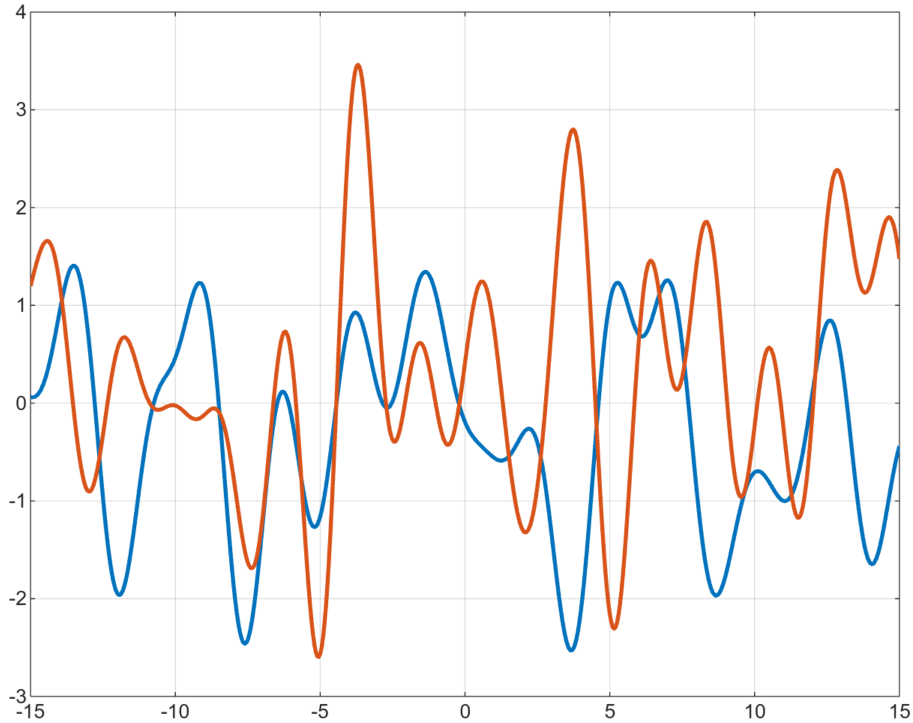
```
ans = 1×2
   601   601
```

```
min(eig(Kytest))
```

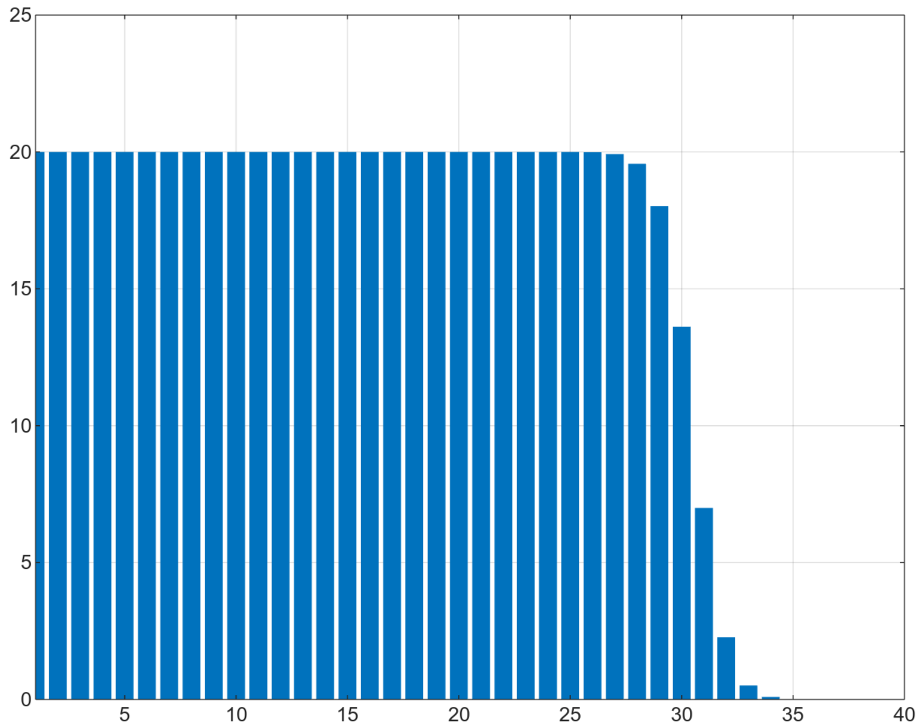
```
ans = 9.4769e-16
```

Example realizations of band-limited signals:

```
RR=mvnrnd(zeros(length(xTest),1),Kytest,2);
plot(xTest,RR,LineWidth=2), grid on
```




```
[V,D]=eig(Kytest);
[esort,idx]=sort(diag(D),'descend');
bar(esort), grid on
xlim([1 40])
```



Interpolation

The famous "sinc" filter or Shannon interpolation formula.

https://en.wikipedia.org/wiki/Sinc_filter

https://en.wikipedia.org/wiki/Whittaker%E2%80%93Shannon_interpolation_formula

```
b=12; %range where available samples are
Xsample=(-b:1:b)';
%f=@(X) (0.6*sin(0.174*pi*X)+.2).*atan(X);
f=@(X) sin(0.4*pi*X);
noisestd=2e-5;
noisevar=noisestd^2;
Nsamples=length(Xsample);

Ysample=f(Xsample)+noisestd*randn(Nsamples,1);
Ytrue=f(xTest);
Ksamples=K(Xsample,Xsample,kappa)
```

```
Ksamples = 25×25
    1.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000 ···
    0.0000    1.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000
   -0.0000    0.0000    1.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000
    0.0000   -0.0000    0.0000    1.0000    0.0000   -0.0000    0.0000   -0.0000
   -0.0000    0.0000   -0.0000    0.0000    1.0000    0.0000   -0.0000    0.0000
    0.0000   -0.0000    0.0000   -0.0000    0.0000    1.0000    0.0000   -0.0000
   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000    1.0000    0.0000
    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000    1.0000
   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000
    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000
      ⋮
```

Note that if samples are separated 1 second, then they are independent; $sinc(N) = 0$ for all integers $|N| \geq 1$.

```
Kysamples=K(xTest,Xsample,kappa);
tmp=Ksamples+noisevar*eye(Nsamples); %samples are contaminated with
measurement noise
min(eig(tmp))
```

```
ans = 1.0000
```

```
CacheINV=pinv(tmp);

%pinv para no invertir valores singulares pequeños... que den
enormes...
%Tolerancias?
max(eig(CacheINV))
```

```
ans = 1.0000
```

```
PredictYmean=Kysamples*CacheINV*Ysample;
PredictYvar=Kytest-Kysamples*CacheINV*Kysamples';
```

As samples (information) are independent from one another, the sample at $t = N$ has a covariance with test point x given by $sinc(x - N)$.

Information from such sample allows a posterior reduction of unexplained variance at x by $sinc(x - N)^2$.

As $sinc(x - N) = \frac{sin(\pi(x - N))}{\pi(x - N)}$, we can write that $(sinc(x - N))^2 = \frac{(\sin(\pi x))^2}{\pi^2(x - N)^2}$ is the variance reduction at $t = x$ given information from a sample at $t = N$. If I have samples at multiple points, said reduction accumulates (remember that `cacheINV` is the identity matrix).

Now, here comes the "Gaussian process" version of Shannon sampling theorem: the following identity holds

$$1 - \frac{\sin(\pi x)^2}{\pi^2} \sum_{N=-\infty}^{+\infty} \frac{1}{(x - N)^2} \ = \ 0$$

At least, so it says Wolfram alpha if you type there **`(Sin[Pi x]^2/Pi^2) Sum[1/(x-N)^2, {N, -Infinity, Infinity}]`** .

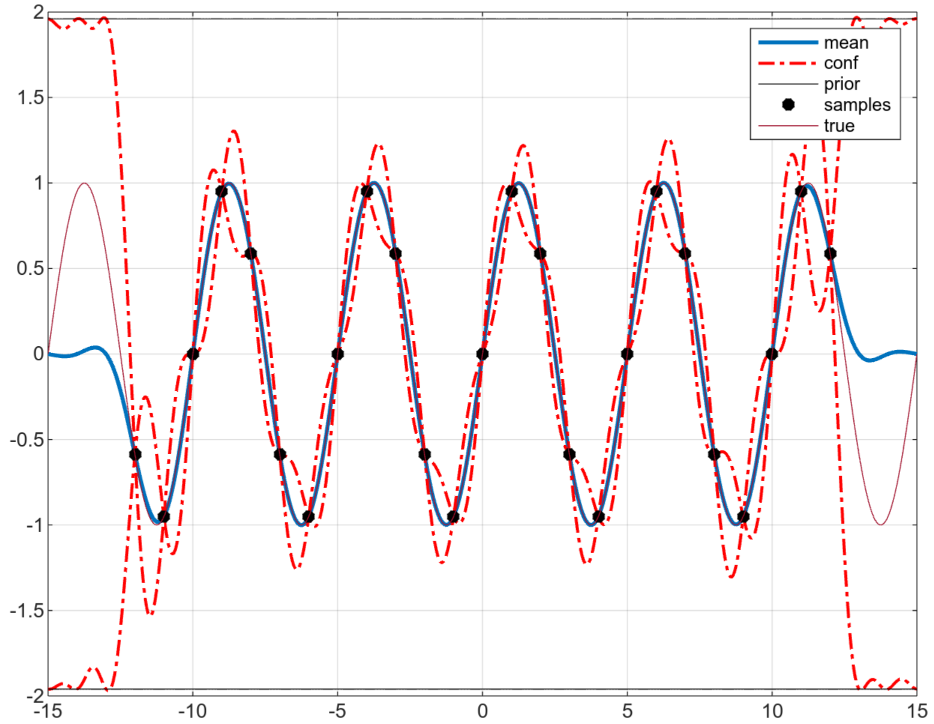
If infinitely many samples are known, then posterior variance reduces to ZERO: the inter-sample values can be perfectly reconstructed.

```
min(diag(PredictYvar)) %check for roundoff/tolerance issues
```

```
ans = 4.0001e-10
```

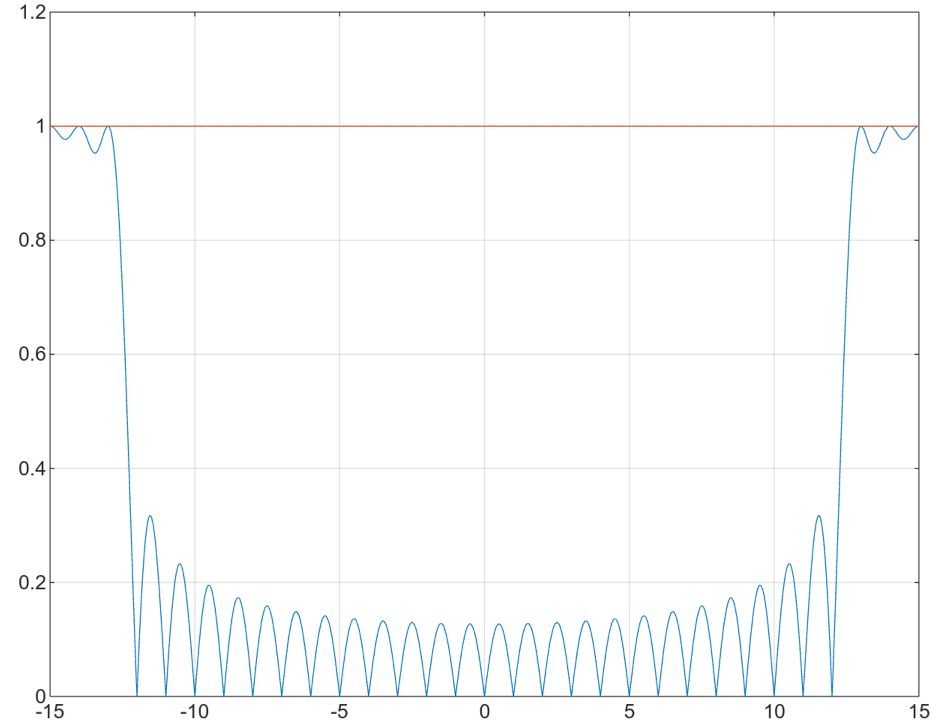
```
PlotWithConfidenceInt(xTest,PredictYmean,PredictYvar,Kytest)
hold on
plot(Xsample,Ysample,'*k',LineWidth=3)
plot(xTest,Ytrue)
```

```
hold off
xlim(xTest([1 end]))
legend("mean","","conf","","prior","samples","true")
```

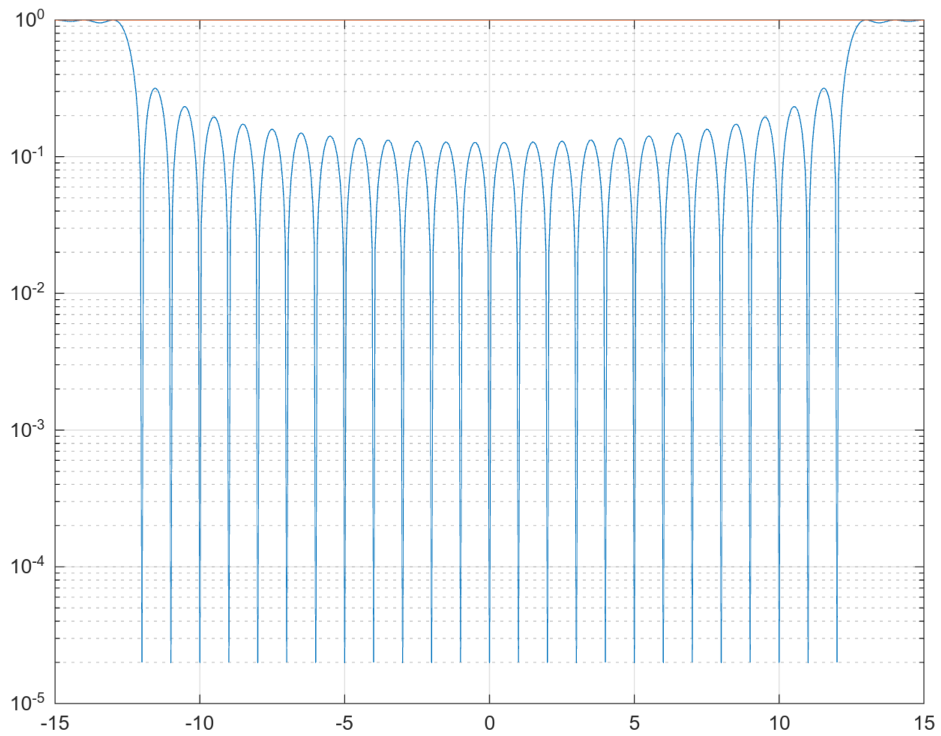


Mean is the reconstruction assuming non-taken samples are zero (expected value). GP stuff allows to give confidence intervals. TRUE is outside with .99*pi???

```
plot(xTest,[sqrt(diag(PredictYvar)) sqrt(diag(Kytest))]) %only
"standard dev" (other figure)
grid on
```




```
semilogy(xTest,[sqrt(diag(PredictYvar)) sqrt(diag(Kytest))]) %only
"standard dev" (other figure)
 grid on
```



Auxiliary functions

```
 function PlotWithConfidenceInt(Xtest,Mean,Var,OLVar)
    stddev=sqrt(max(0,diag(Var)));
    stddevOL=sqrt(max(0,diag(OLVar)));
    plot(Xtest,Mean,LineWidth=2), hold on
    plot(Xtest,[Mean-1.96*stddev
Mean+1.96*stddev],'r-.',LineWidth=1.5), grid on
    plot(Xtest,[-1.96*stddevOL 1.96*stddevOL],'k')
    hold off
 end


function M=K(x1,x2,KernelFunction)
%Kernelfunction returns a SCALAR
% STATIONARY KERNEL, only DISTANCE as argument
%data samples in x1 and x2 are ROWS
N1=size(x1,1);
N2=size(x2,1);
M=zeros(N1,N2);
for i=1:N1
    for j=1:N2
        M(i,j)=KernelFunction(norm(x1(i,:)- x2(j,:)));
    end
end
end
```