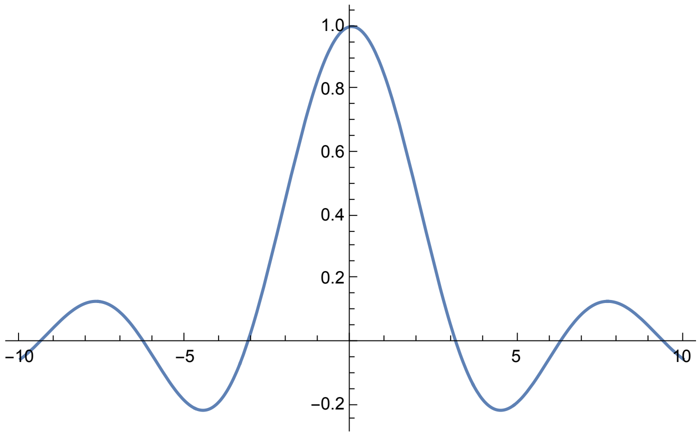
# Commande Numérique des Systèmes Dynamiques
# Echantillonnage et Reconstruction.
# Exemple du phénomène de Gibbs.
Ph. Müllhaupt

Echantillonnage et reconstruction d'un signal avec 4 cosinus en rapport harmonique et pondérés, tronqués par une fenêtre rectangulaire de t qui vaut 0 à t valant 1. Le signal est échantillonné puis reconstruit à l'aide du sinus cardinal...

Avant toute chose, examinons la fonction clé de la reconstruction... le sinus cardinal...

In[3]:=
```
Plot[Sinc[t], {t, -10, 10}]
```

Out[3]=

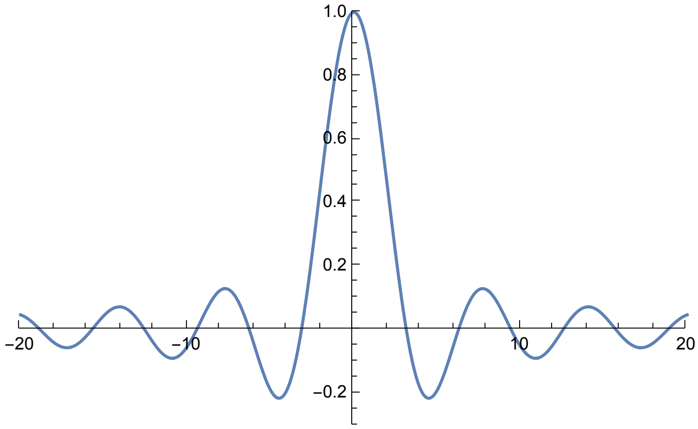



In[4]:=

... changeons un peu le domaine pour dessiner un peu plus ...

In[6]:=
```
Plot[Sinc[t], {t, -20, 20}, PlotRange → {{-20, 20}, {-.3, 1}}]
```

Out[6]=




Remarquez que tous les zéros sont séparés de Pi, et qu' à zéro cela donne 1

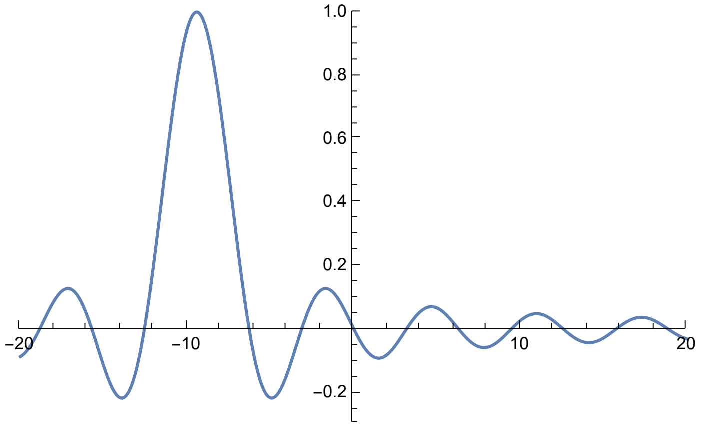
In[8]:=
```
Sinc[Pi] // N
```

Out[8]= `0.`

... en 3 Pi sela passe aussi par zero ...

In[10]:=
```
Plot[Sinc[t + 3 Pi], {t, -20, 20}, PlotRange → {{-20, 20}, {-.3, 1}}]
```

Out[10]=

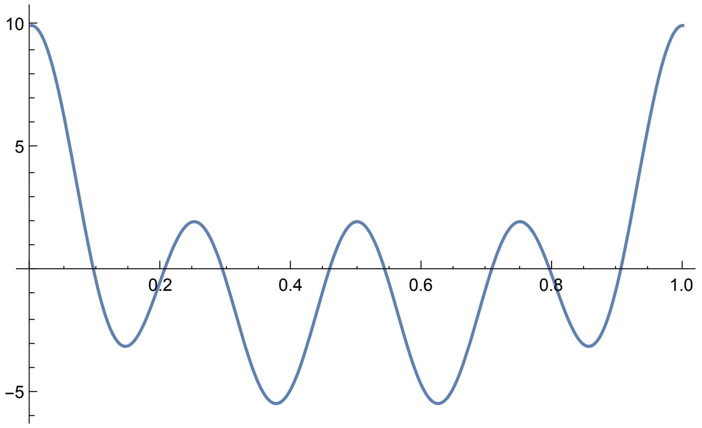


In[11]:=
```
fta = 3 Cos[2 Pi t] + 2 Cos[4 Pi t] + 1 Cos[6 Pi t] + 4 Cos[8 Pi t];
```

Voici le signal à échantillonner et à reconstruire :

In[13]:=
```
Plot[fta, {t, 0, 1}]
```

Out[13]=



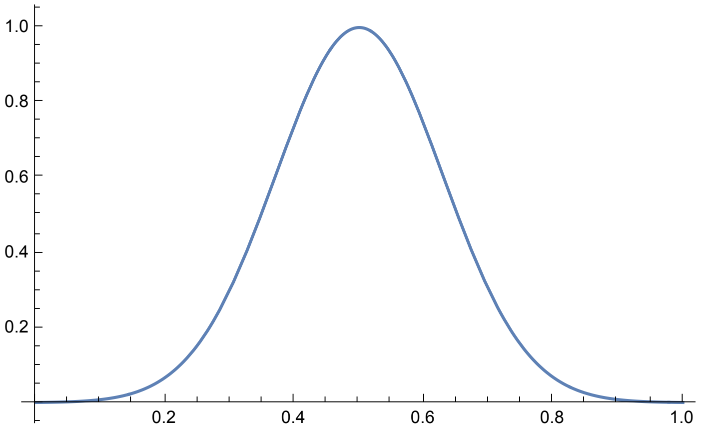
On constate que le signal est discontinu à gauche de t = 0 et à droite de t = 1, par rapport à la valeur constante 0.

Pour adoucir le signal sur les bords de la fenêtre, on utilise un pondérage gaussien:

In[16]:=
```
win = Exp[-30 (t - 0.5) ^2];
```

In[17]:=
```
Plot[win, {t, 0, 1}]
```

Out[17]=



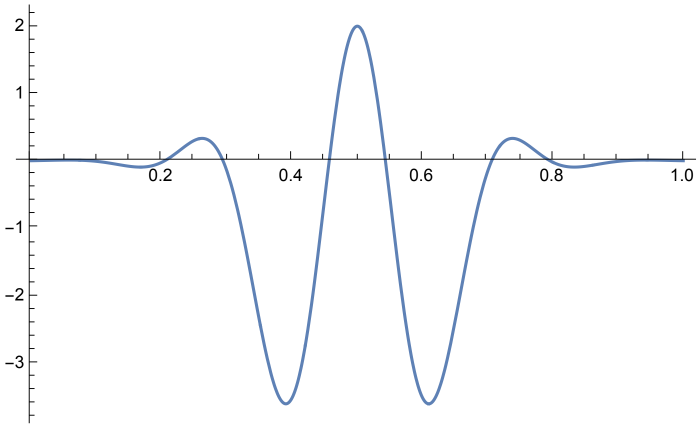
La fonction gt suivante est le résultat de ce pondérage :

In[19]:=
```
gt = fta * win;
```

In[20]:=
```
ft = fta;
```

In[22]:=
```
Plot[gt, {t, 0, 1}]
```

Out[22]=




On constate que le signal est continue à gauche et à droite de la fenêtre (en considérant que ce signal est prolongé à gauche et à droite par la valeur constante 0)

Soit M qui désigne le nombre d' échantillons :

In[24]:=
```
M = 100;
```

Comme la durée du signal est 1 on trouve le période d' échantillonnage hh

In[26]:=
```
hh = 1 / M // N;
```

la suite des instants d'échantillonnage est tk

In[28]:=
```
tk = Table[i / M, {i, 1, M}] // N;
```

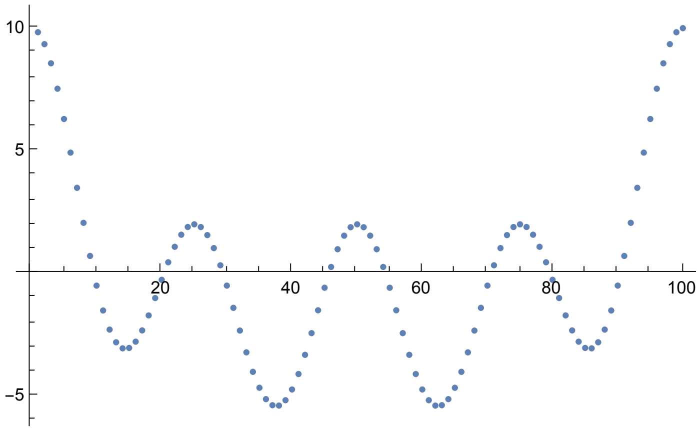
... et les échantillons sont fk et gk...

In[30]:=
```
fk = ft /. {t → #} & /@ tk // N;
```

In[31]:=
```
gk = gt /. {t → #} & /@ tk // N;
```

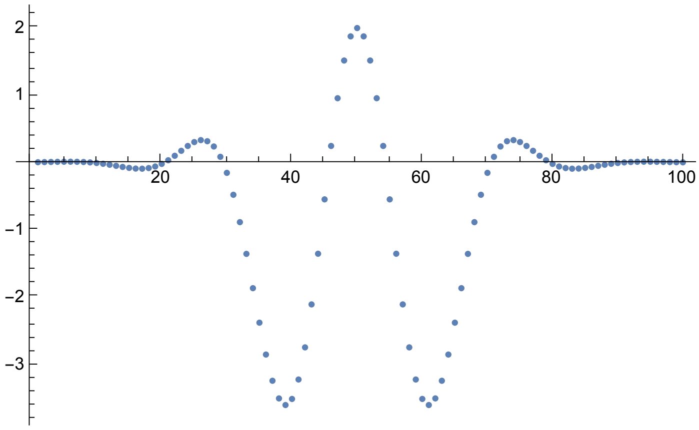
In[32]:=
```
ListPlot[fk]
```

Out[32]=



In[33]:=
```
ListPlot[gk]
```

Out[33]=




On construit les sinc décalés sur chaque instant d'échantillonnage

In[35]:=
```
sincs = Sinc[ t Pi / hh - Pi / hh #] & /@ tk;
```

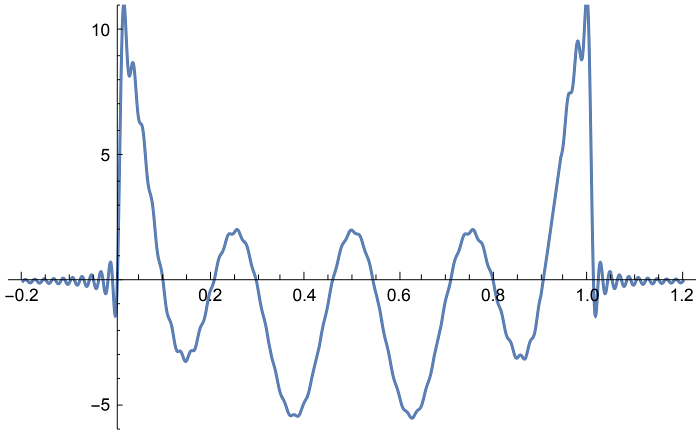
econstr est la variable qui correspond aux échantillons pondérés par les sinc

In[42]:=
```
reconstrF = MapThread[#1 #2 &, {sincs, fk}];
```

... et il ne reste plus qu'à inspecter ce que cela donne ...

In[43]:=
```
Plot[Fold[#1 + #2 &, 0, reconstrF], {t, -0.2, 1.2}, PlotRange → {-6, 11}]
```

Out[43]=



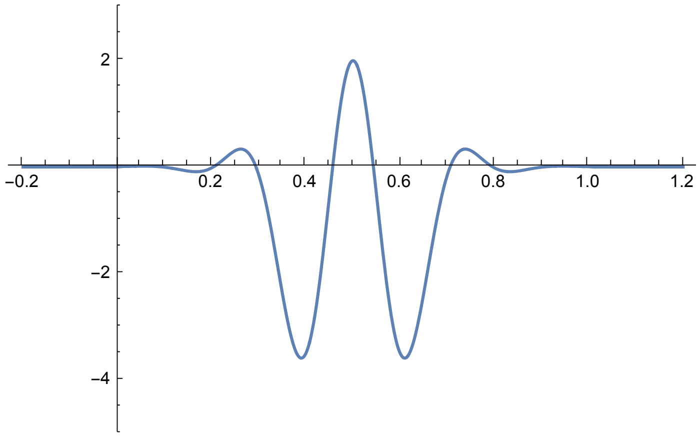

L'approximation ne provient pas du mauvais choix de la période d'échantillonnage, mais de la nature tronquée du signal de départ: il ne part pas du temps - infini et il ne finit pas au temps + infini. Il n'a donc pas un contenu fréquentiel à bande limitee. Il ne  satisfera jamais aux conditions du théoreme d'échantillonnage !

En ce qui concerne le signal g qui a été adouci par la fonction gaussienne, la reconstruction ne pose aucune difficulté et le phénomène oscillatoire sur les bords de la fenêtre n'est pas perceptible.

In[44]:=
```
reconstrG = MapThread[#1 #2 &, {sincs, gk}];
```

In[47]:=
```
Plot[Fold[#1 + #2 &, 0, reconstrG], {t, -0.2, 1.2}, PlotRange → {-5, 3}]
```

Out[47]=




- REMARQUE IMPORTANTE:

DANGER : Sinc[x] de Mathematica = sinc(pi*x) de Matlab et SysQuake