-final rev0.1-

Šumová analýza synchronní kumulace

Petr Sládek, CTU FEE
sladek@asix.cz; www.smishek.com

Měřený signál $s(t)=p(t)+n(t)$ definujeme užitečným periodickým signálem $p(t)$ a aditivním šumem $n(t)$ s hustotou pravděpodobnosti $d(n,params)$; *„params“* je vektor prametrů rozdělení (normální, binomické,...).

Kumulační technika zvyšování SNR využívá nulové či konstantní střední hodnoty aditivního šumu, průměruje D-krát[1] opakující se periodické vzorky s periodou T (to bývá synchronizační signál). Díky nekorelovanému šumu $n(t)$ dostáváme:

$$s_{SNRE}(t)=\frac{1}{D}\sum_{k=1}^{D} p(t+kT)+n(t+kT)=\frac{1}{D}\left(D\,p(t)+\sum_{k=1}^{D} n(t+kT)\right)=\cdots$$

$$\cdots=p(t)+\frac{1}{D}\sum_{k=1}^{D} n(t+kT)=p(t)+\tilde{n}(t)$$

Aplituda šumu je po filtraci rovna aritmetickému průměru $n(t+kT)$ z D vzorků. Pokud uvážíme, že šum má nulovou nebo kostantní známou střední hodnotu, je nekorelovaný, dojde ke snížení amplitudy šumu.

Zkoumejme nyní jen „nový“, průměrovaný šum $\tilde{n}(t)=\frac{1}{D}\sum_{k=1}^{D} n(t+kT)$. Hledáme kolikrát se zmenší $var\,N$ při D-násobné filtraci. Využitím $var\sum X_i=\sum var\,X_i$ a $var\,aX=a^2\,var\,X$ pro nekorelované náhodné veličiny dostáváme

$$var\,\tilde{N}=\frac{1}{D^2}\sum_{k=1}^{D} var\,N=\frac{1}{D}\,var\,N$$

Pro normální rozdělení pak platí $\tilde{\sigma}=\frac{1}{\sqrt{D}}\sigma$, pro symetrické rovnoměrné rozdělění o „šířce“ a platí $var\,\tilde{N}=\frac{1}{D}\frac{1}{12}2\,a^2=\frac{1}{6\,D}a^2\rightarrow$ $\tilde{a}=\frac{1}{\sqrt{D}}a$. Z uvedeného vyplývá, že se SNR zvyšuje $\sqrt{D}$ krát při absolutně přesné synchronizaci repeticí signálu $p(t)$ [2]. Tato analýza se týkala spojité kumulace prakticky nerealizovatelné (nekonečná paměť), pokud je ale šum frekvenčně omezen shora, lze kumulaci provést číslicově. Důležité je, že $\sqrt{D}$ závislost zvýšení SNR platí i v diskrétním případě. Toto tvrzení bylo ověřeno numericky – viz. příloha 1.

Časová nestabilita (jitter) synchronizačního signálu se projeví v překrývání vzorků (synchronous sample jitter):

1 Depth, hloubka či počet bufferů (akumulátorů dat) pro průměrování
2 Jiné odvození naleznete v [1] – Kumulační metody zvýrazňování signálu.

Sync No Jitter

sample#								
buffer0	1	2	3	4	5	6	7	8
buffer1	11	12	13	14	15	16	17	18
buffer2	21	22	23	24	25	26	27	28

Sync. Jitter

1	2	3	4	5	6	7	8	
8	11	12	13	14	15	16	17	18
21	22	23	24	25	26	27	28	31

missing values – default 0

Zahrnutím jitter synchronizačního signálu se změní měřený signál, tentokráte již vzorkovaný na $s(i)=p(i)+\tilde{n}(i)+\tilde{j}(i)$. Složka $\tilde{j}(i)$ [3] je šum způsobený překrýváním sobě neodpovídajících si vzorků. Pro jednoduchost jsou $i \sim iT_{samp}$, lze si pod tím představit *i*-té vzorky (tj. obsah paměti) nebo iT_{samp} vzorky signálu v reálném čase. Jelikož je $\tilde{n}(i)$ nekorelovaný, lze vliv jitter sync. signálu zanedbat a šum $\tilde{n}(i)$ považovat za nezávislý na $\tilde{j}(i)$. Oba šumy budeme tedy zkoumat nezávisle na sobě. Pro každé dva repetiční vzorky (např. obsah buffer0, buffer1) platí

$$j(i)=p(i)-p(i+j_{s01})$$

kde j_s je „vzorková odlehlost“ $p(i)$ bufferu 0 a 1, tj. o kolik vzorků se signál posunul díky jitter. Signál $p(i)$ musí být samozřejmě periodický pro kT . Šum j_s je tedy diskrétní náhodná veličina nabývajících diskrétních hodnot <-N,N> způsobující „rozmazání“ signálu v časové ose.

Z výše uvedeného je zřejmé, že propagace j_s do výsledného signálu $s(t)$ SNR Enhanced závisí na průběhu „čistého“ měronosného signálu $p(t)$. Pokud bude $s(i)$ konstanta neprojeví se vůbec, pokud bude $p(i)$ „toggle“ *-c* každý sudý a *c* každý lichý vzorek, projeví se velmi významě, až destrukcí měronosného signálu. Intuitivně bez dalších „vědeckých“ odvození vidíme, že pro efektivní potlačení jitter sync. signálu je nutné mnohonásobě převzorkovat (desítky až tisíce) měronosný signál a tím dosáhnout co nejhladšího (nejploššího) průběhu $p(i)$. Tato metoda je efektivní pokud je jitter srovnatelný se vzorkovacím intervalem, pokud je vzorkovací interval mnohonásobě kratší než jitter, přestává mít další zvyšování vzorkovací frekvence smysl.

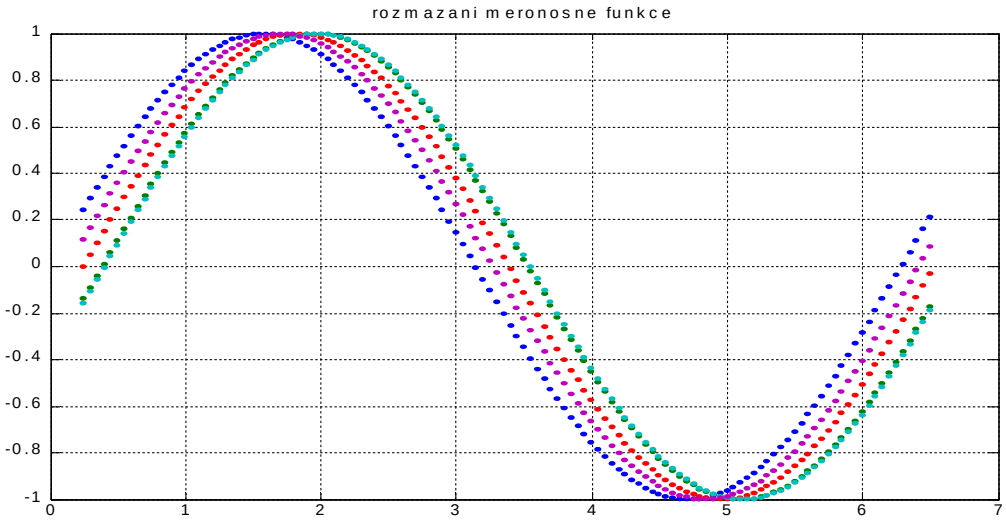
Pro každou třídu funkcí $p(i)$ je nutné určit novou hustotu pravěpodobnosti, při známé hustotě $j(i)$

- nulová pro necitlivé funkce na jitter,
- kalkuluje s vzorkovací frekvencí – tj. odráží rychlosti změn mezi sousedími vzorky.

Uvádím ilustrační obrázek, co se děje v tomto modelovém případě:

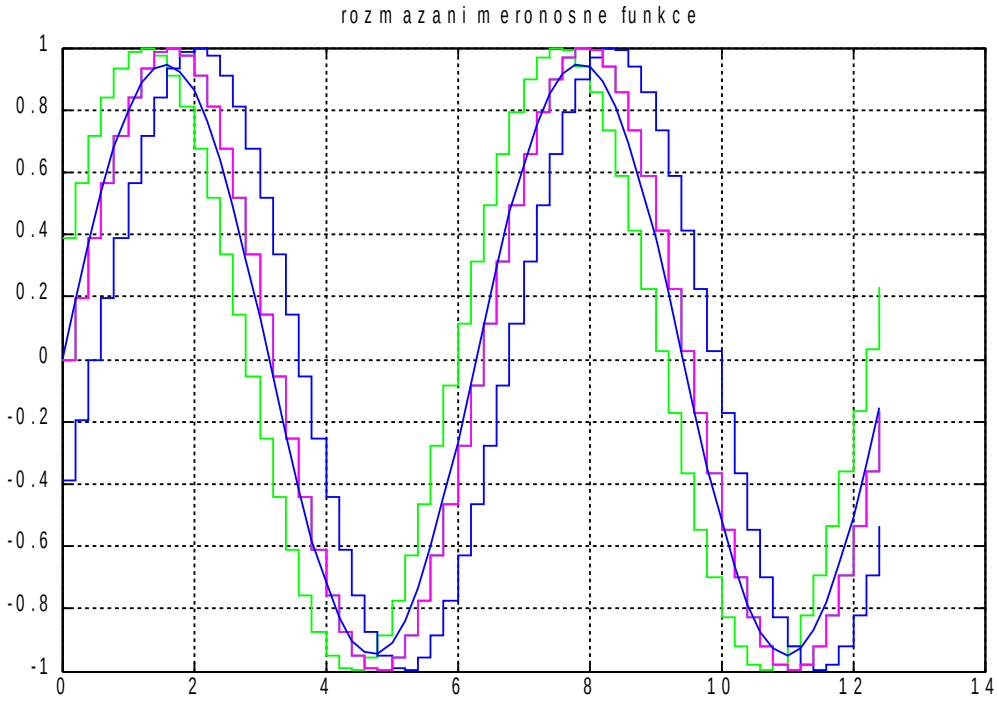
3 *j* s vlnkou má stejný význam jako *i* s vlnkou, jde o průměr ze všech bufferů

Princip:



Obr 1.

Skutečnost v bufferech (odlehlost je minimálně 1 sample).

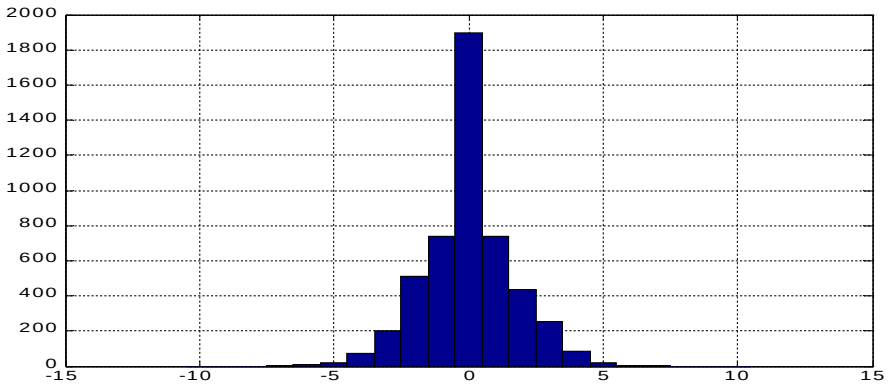


Obr 2. Pozn. hladký, modrý signál je průměrná hodnota z 10, zobrazeno pouze 3.

Husota pravděpodobnosti pro tento ukázkový případ byla zvolena je diskrétní „normální“

```
dt= fix(normrnd(0,2,[D,1])'):
```

Obr 3. Hodnoty na vodorovné ose udávají posun ve vzorcích tj. Tjitter/Tsamplovací.



Výsledný šum $\tilde{j}(i)$ lze modelovat jako D-násobný průměr rozdílu všech posunutých vzorků v bufferech vůči referenčnímu vzorku. Upozorňuji, že toto je zjednodušený model, který postihuje jen malé amplitudy $j_s(i)$, protože není jasné jakým způsobem se budou v praxi extrapolovat chybějící vzorky a co se bude dělat s přebývajícími vzorky (vhodnou technikou by je šlo totiž použít pro doplnění chybějících v bufferech. Pokud známe přesnou periodu sync. signálu (např. průměrovaná perioda sync. signálu) není veliký problém automaticky korigovat „shift" (jitter) v bufferech a tím šum eliminovat. Tento model počítá s tím, že se nic neeliminuje a měronosná fukce vykazuje mezi dílčími buffery pouze drobné fázové posuny. Bude tedy dávat hoší výsledky pokud se nepoužije autokorekce jitter a o málo lepší než když se chybějící vzorky nahradí nulou a přebývající se odseknou. Srovnatelné pak, pokud se použije extrapolace – dopočtení chybějících vzorků z jedné periody. Obecně tedy dostáváme

$$\tilde{j}(i)=\frac{1}{D-1}\sum_{k=0}^{D-2}\left(p\left(\frac{T}{N}(i+j_{i\div N-k})\right)-p\left(\frac{T}{N}i\right)\right),\quad \text{v indexu } j_{i\div N} \quad =i\div N \text{ celočíselně.}$$

T je perioda měronosného signálu, N množství vzorků na periodu, $p(ki)$ je samplovaná měronosná funkce. Protože zadání požaduje zkoumat sinusovku jako $p(ki)$, model šumu vypadá následovně

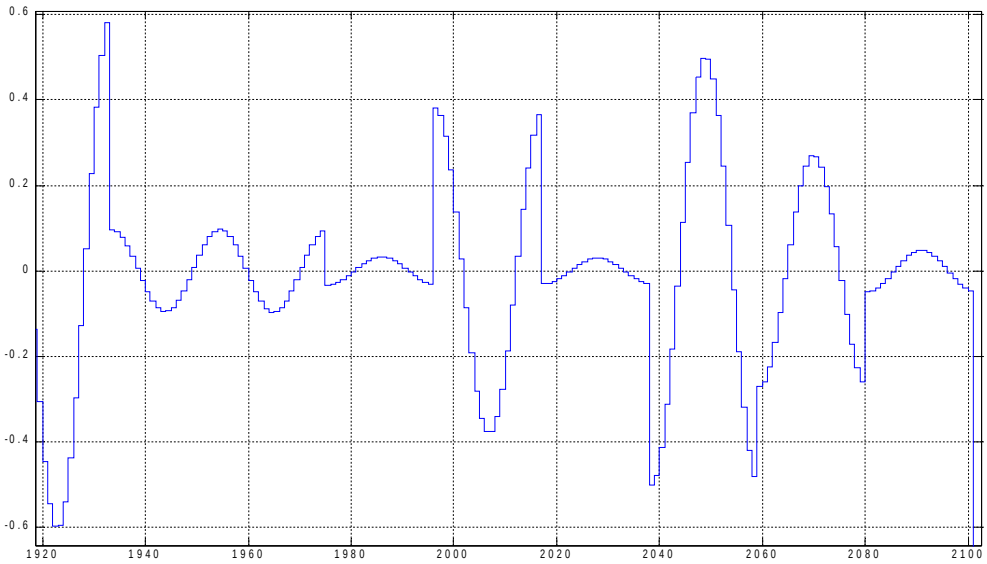
$$\tilde{j}(i)=\frac{1}{D-1}\sum_{k=0}^{D-2}\left(\sin\frac{2\pi}{N}(i+j_{i\div N-k})-\sin\frac{2\pi}{N}i\right) \quad .$$

Protože uvažujeme $j_{i-k}\ll N$ lze pomocí $\sin(\alpha+\beta)=\sin\alpha\cos\beta+\cos\alpha\sin\beta$, $\sin\alpha\approx\alpha \quad \alpha\to 0$ a $\cos\alpha\approx 1 \quad \alpha\to 0$ ukázat, že

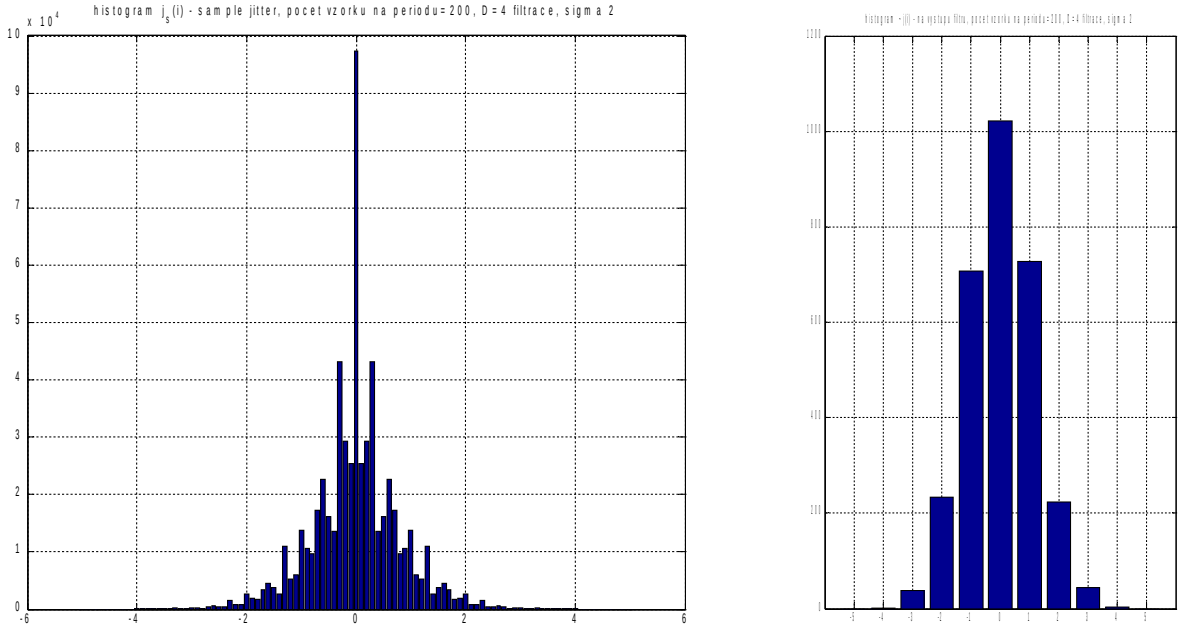
$$\tilde{j}(i)\approx\cos\left(\frac{2\pi}{N}i\right)\frac{2\pi}{N}\frac{1}{D-1}\sum_{k=0}^{D-2}j_{i\div N-k}$$

pro D=1 (žádná kumulace) musí být definitoricky $j(i)=0$, protože šum vzniká průměrováním vzájemě posunutých vzorků. Jitter $j_{i\div N}$ nabývá diskrétních hodnot z oboru přirozených čísel, ale

$\tilde{j}(i)$ nemusí i když je diskrétní. Např. pro D=3 může nabývat hodnot -1.5, 0.5, 0..[4] Má takovýto průběh (ilustrace):

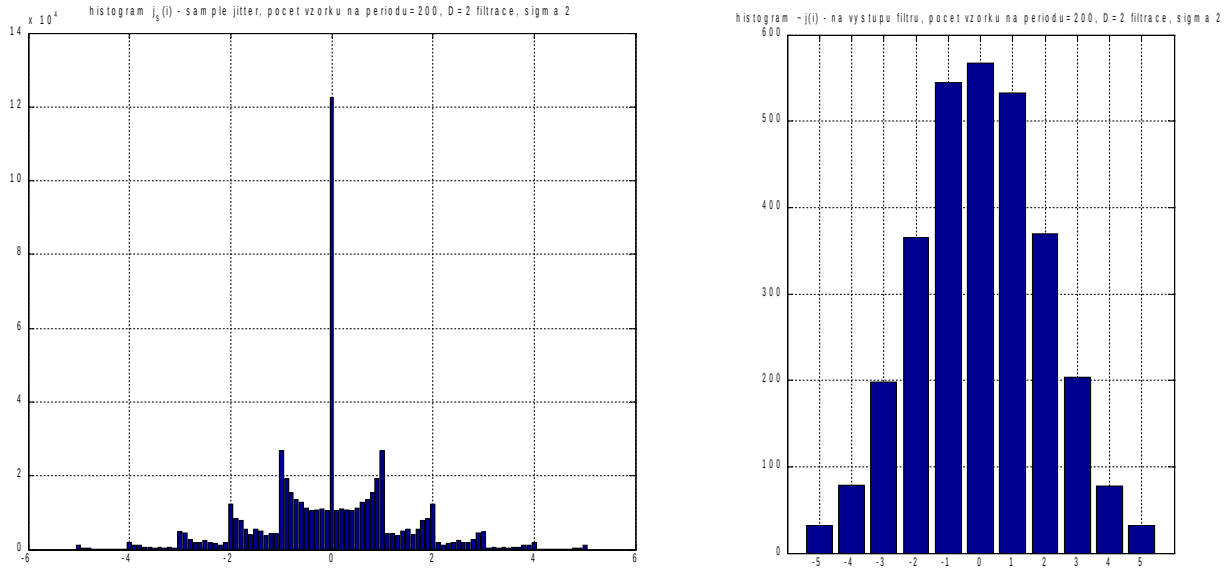


„Hrubou silou“ (výpočtem) jsem získal husotu pravěpodobnosti, pokud má $j_{i \div N}$ normální rozložení (bohužel vypočítat hustotu je značný problém, konzultoval jsem tento statistický oříšek s RNDr. L. Nentvichem):

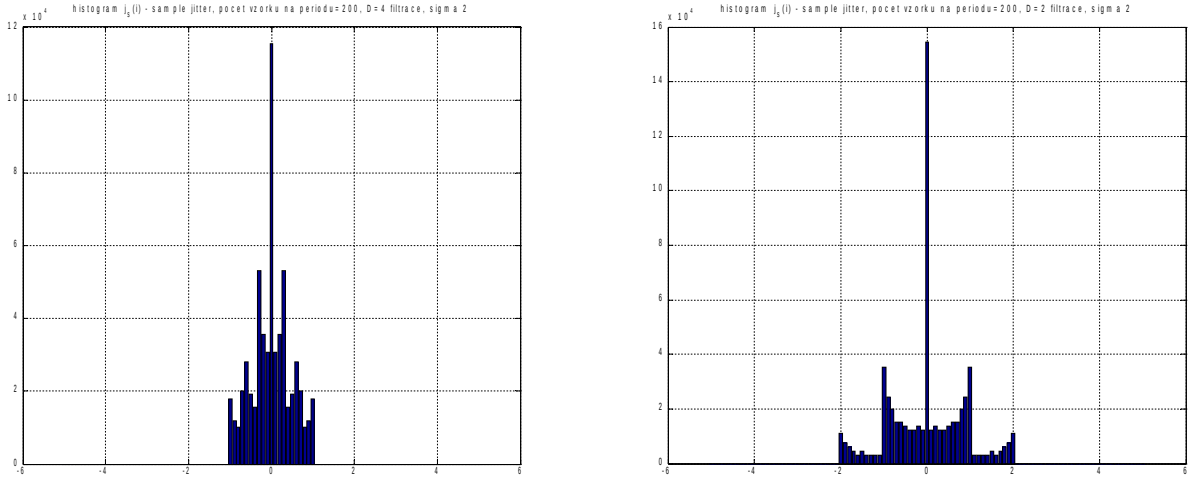


Obr. 5 Hustota pro šum $\tilde{j}$ a diskrétní hodnoty normální $j_{i \div N}$ vpravo N=200, **D=4**.

4 To je zohledněno v m-file getQuasinHist(): yrnd=(1/(D-1)).*round((D-1).*normrnd(0,sigma/sqrt(D-1),[3000 1]));



Obr.6 Hustota pro šum $\tilde{j}$ a diskrétní hodnoty normální $j_{i\div N}$ vpravo N=200, **D=2**.



Obr.7 To samé ve stejném pořadí s rovnoměrným rozložením $j_{i\div N}$.

Tento „kvazináhodný“ šum se přičítá k výslednému signálu společně s aditivním šumem přicházejícím ze senzoru $s(i)=p(i)+\tilde{n}(i)+\tilde{j}(i)$. Domnívám se, že analytický výpočet takových hustot je úkol pro zkušeného matematika, proto jsem jak hustoty tak jejich rozptyly počítal pomocí Matlabu („hrubou silou“). Pro získání hustoty jsem napsal funkci

`function [Hj H] = getQuasinHist(gauss,N,p,D,Jinterval,Vinterval)`[5]

gauss =1 normání, =0 rovnoměrné; N počet vzorků na periodu; p parametr hustoty, D počet průměrování, Jinterval, Vinterval interval pro histogram jitter a hodnot samotného signálu.

5 viz příloha 3

Celkový výpočet SNR provádí skript snrej.m[6], který projde množinu hodnot N a D

```
selN = [10 20 50 100 200];
selD = [2 3 4 5 10 20 50 100 200];
```

při němž automaticky adjustuje šířku historamu tak, aby nepřetékal a nebo nebyl využit jeho rozsah (to způsobuje obrovské chyby, horší než přetečení). Adjustace intervalů histogramu se provádí napůl heuristicky, celý soubor se neprochází:

```
sVinterval = (Vinterval/(sqrt(sqrt(N))*sqrt(D-1)*(D-1)));
sJinterval = Jinterval*sqrt(N);
```

Protože getQuasinHist() vrací hustoty dle zadaného jitteru v samplech, nikoliv času je nutné tuto hodotu upravit, podle aktuálního převzorkování N:

```
sp=p*(N/(2*pi));
```

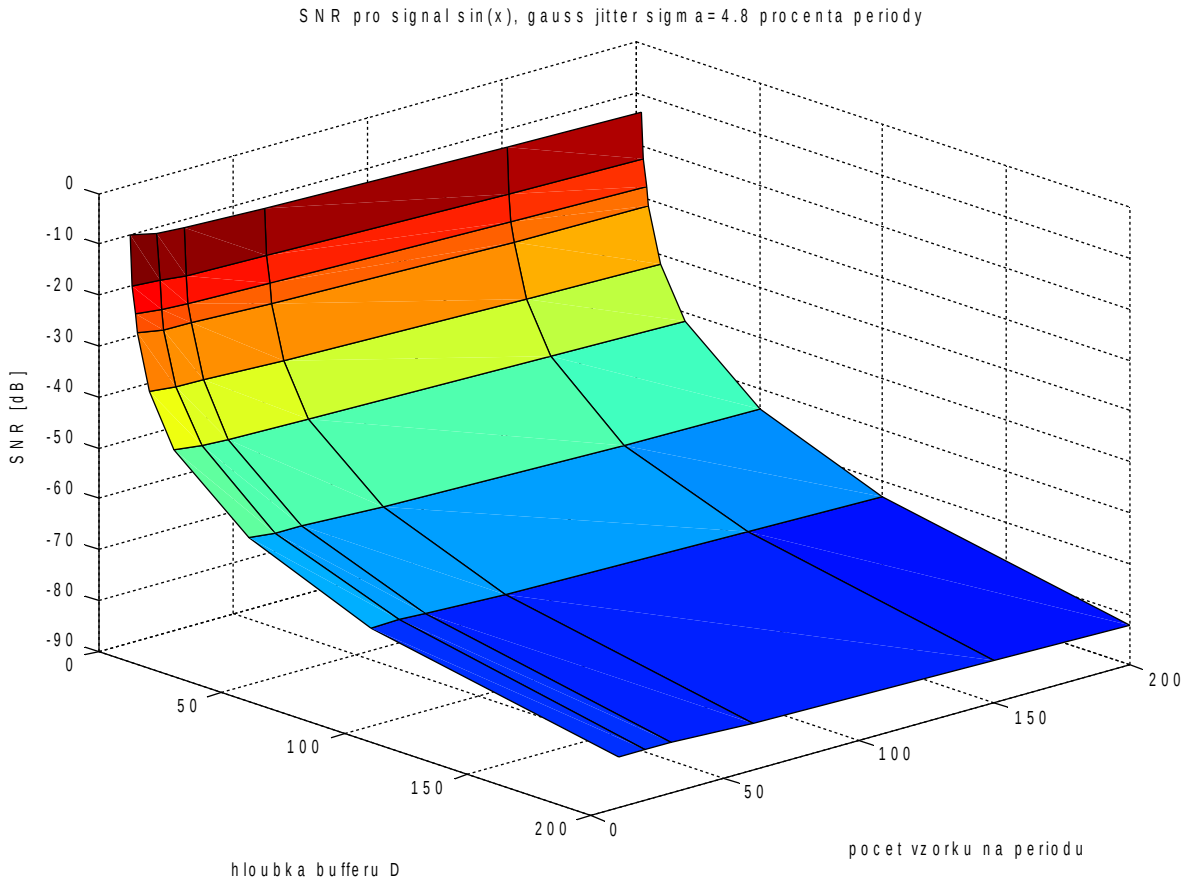
získáme aktální hustotu

```
[Hj H] = getQuasinHist(GAUSS,N,sp,D,sJinterval,sVinterval);
```

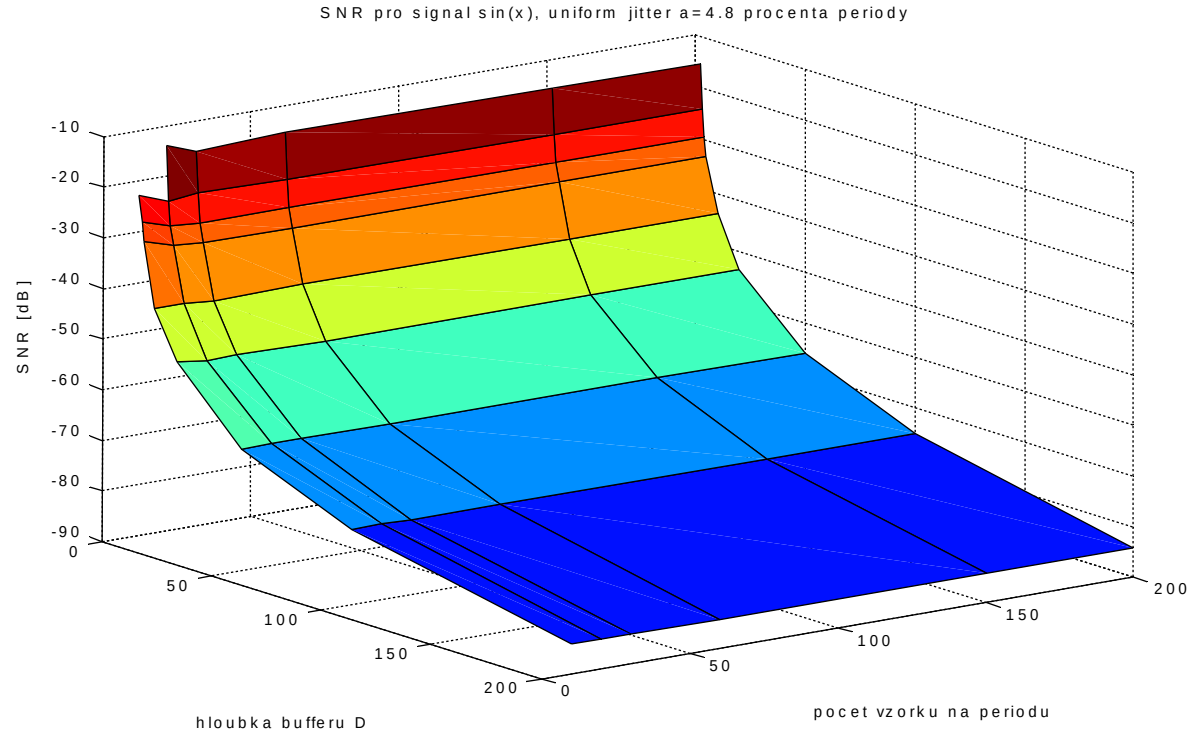
vypočítáme rozptyl

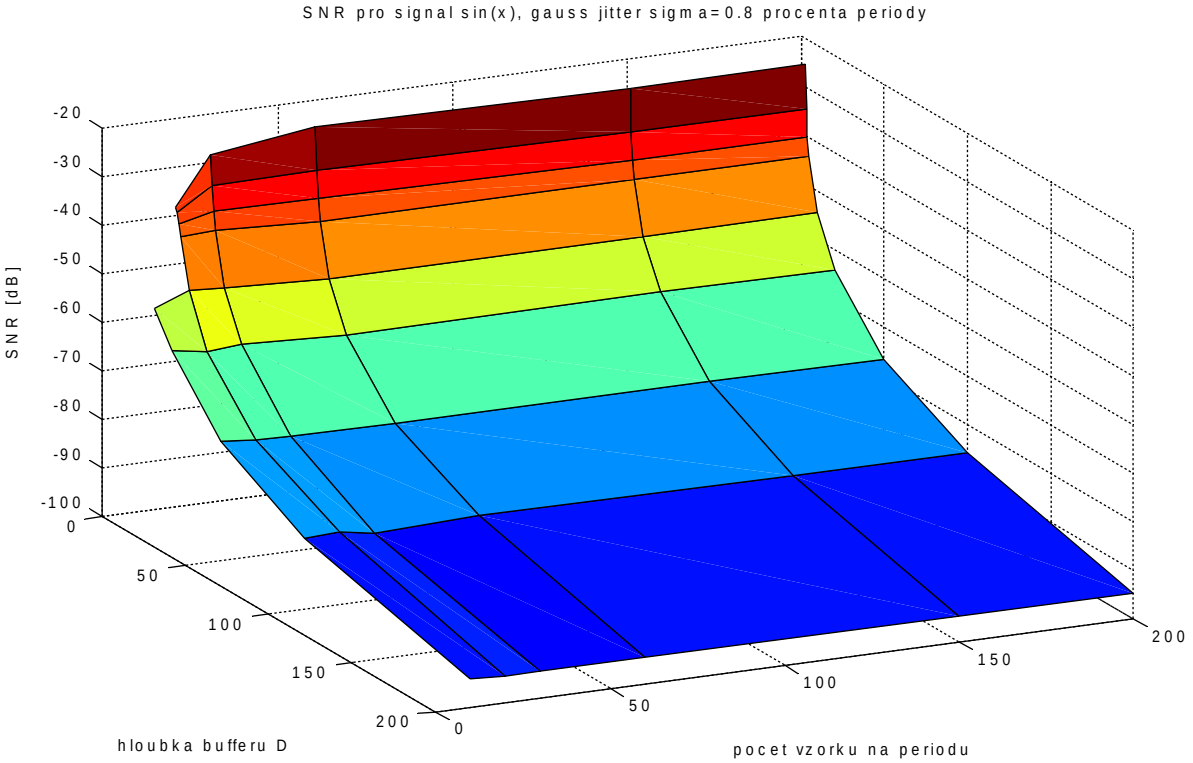
```
varJ=(sVinterval.*sVinterval)*H';
```

Výsledky výpočtů:



6 viz příloha 2

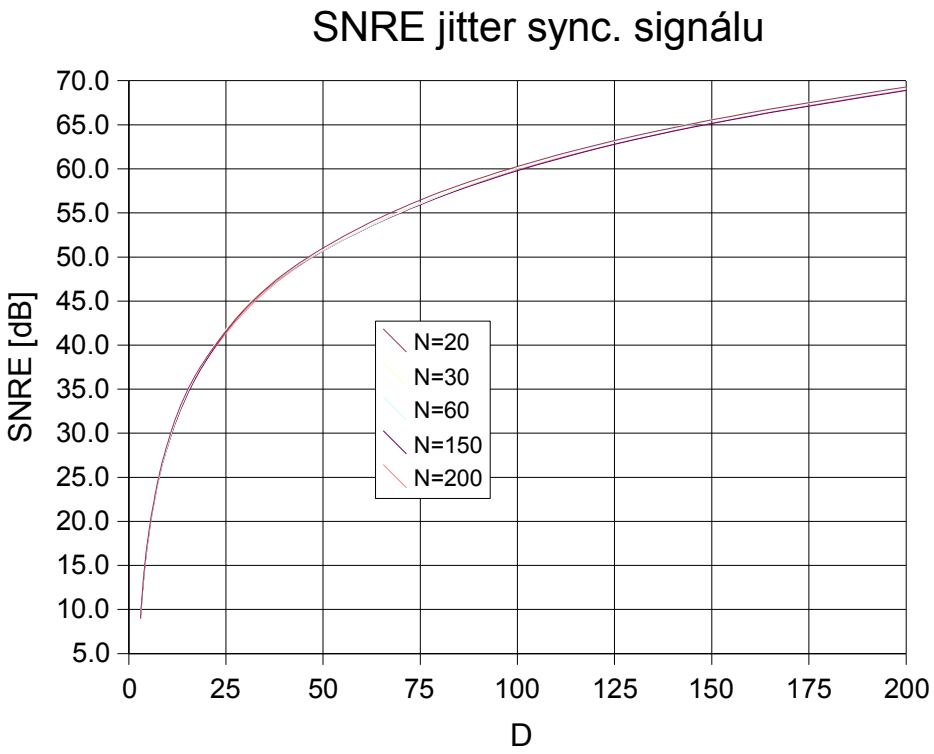

SNR pro signal sin(x), uniform jitter a=4.8 procenta periody
SNR [dB]
-10
-20
-30
-40
-50
-60
-70
-80
-90
0
50
100
150
200
hloubka bufferu D
0
50
100
150
200
pocet vzorku na periodu


SNR pro signal sin(x), gauss jitter sigma=0.8 procenta periody
SNR [dB]
-20
-30
-40
-50
-60
-70
-80
-90
-100
0
50
100
150
200
hloubka bufferu D
0
50
100
150
200
pocet vzorku na periodu

Na posledním grafu je trochu patrné zlepšení SNR při menším množství vzorků – ano, je to logické, protože se sníží citlivost na rušení fáze, ale v praxi je potřeba spíš desítky až stovky vzorků na obecnější funkci než je sinus.

SNRE jako rozdíl SNR_před_filtrcí – SNR_po_filtraci aditviního šumu je $20\log\sqrt{D}=10\log D$. Pro signály s normálním rozložením platí $\tilde{\sigma}=\frac{1}{\sqrt{D}}\sigma$ a s rovnoměrným rozložením $\tilde{a}=\frac{1}{\sqrt{D}}a$.

SNRE jitter sync signálu jde odvodit z rozdílu SNR_D=2 – SNR_po_filtraciD>2. Jitter šum vzniká nedokonalým zasynchronizováním až v kumulátoru, samotný signál nic takového neobsahuje. Toto tvrzení lze jednoduše ověřit výpočtem několika hodnot amplitudy šumu a výsledného SNR[7]. Průběh SNRE se blíží jen velmi přibližně $10\log\sqrt{D}\,D^2$.



Tato závislost platí pro jitter přibližně 20x a vícekrát nižší než je perioda signálu. Mezi normálním a rovnoměrným rozložením není znatelný rozdíl (v rámci přesnosti výpočtu).

Literatura:

Číslicová filtrace, analýza a restaurace signálů, Jiří jan. VUT Brno. nakl VUTIUM, ISBN80-214-1558-4

7 Viz příloha 4

Příloha 1

4x průměrovaný šum
stredni hodnota x_i je: 9.976345e-004
stredni hodnota kvadratu x_i je: 2.511952e-001
odhad rozptylu hodnot s: **5.011928e-001**

neprůměrovaný šum
stredni hodnota x_i je: -3.991835e-003
stredni hodnota kvadratu x_i je: 1.000156e+000
odhad rozptylu hodnot s: **1.000070e+000**

skript testfilt.m:

```
wav1 = normrnd(0,1,[50000 1]);
wav2 = normrnd(0,1,[50000 1]);
wav3 = normrnd(0,1,[50000 1]);
wav4 = normrnd(0,1,[50000 1]);

%wav = (wav1+wav2+wav3+wav4)./4;
wav=wav1;

%stredni hodnota
mu = mean(wav);
fprintf('stredni hodnota x_i je: %d \n',mu);

%odhad rozptylu:
av2 = mean(wav.*wav); %dot product = x_i^2
fprintf('stredni hodnota kvadratu x_i je: %d \n',av2);

s = sqrt(av2 - power(mu,2));

fprintf('odhad rozptylu hodnot s: %d \n',s);
```

Příloha 2

```
%vypocet SNR jitter

selN = [20 30 60 150 200];
selD = [2 3 4 5 10 20 50 100 200];
%selN=[200];
%selD=[4];
GAUSS=1;
p = 0.4;

Jinterval=-2:0.2:2;
Vinterval=-1.5:0.05:1.5;

lenN=length(selN);
lenD=length(selD);
selVPS = zeros(lenN,lenD);
selMPS = zeros(lenN,lenD);

for iN=1:lenN,
 for iD=1:lenD,
N=selN(iN);
D=selD(iD);
%zjemnovani intervalu (kvuli filtraci) a vypocet histogramu:
sVinterval = (Vinterval/(sqrt(sqrt(N))*sqrt(D-1)*(D-1)));
sJinterval = Jinterval*sqrt(N);
%protoze p udava jitter v samplech je nutne hodnotu upravit, tak aby
%byla vyjadrena v case tj na periodu 2pi (tj. konstatna nezavisle na sample rate).
%jitter_cas = 2*pi*(jitter_samples/N)
sp=p*(N/(2*pi));
[Hj H] = getQuasinHist(GAUSS,N,sp,D,sJinterval,sVinterval);

%vypocet rozptylu:
varJ=(sVinterval.*sVinterval)*H';

%kontrola overflow/underflow
if (sum(H(1:5))>0.005)
  fprintf('Histogram bounce overflowed\n!');
end;
%prozkoumat okoli jestli je nula.
x=floor(length(H)/5);
if (sum(H(x:(x+5)))<0.01)
  fprintf('Histogram bounce underflowed\n!');
end;


if (1)
figure(2); clf;
bar(sJinterval,Hj); grid on;
title(sprintf('histogram ~j(i) - na vystupu filtru, pocet vzorku na periodu=%d,
D=%d filtrace, sigma %d',N,D,p));
figure(3); clf;
bar(sVinterval,H); grid on;
title(sprintf('histogram j_s(i) - sample jitter, pocet vzorku na periodu=%d, D=%d
filtrace, sigma %d',N,D,p));
```

```
end

selVPS(iN,iD)= varJ;

 end;
end;
%predpoklad, ze se funkce RMS moc nemeni a lze linearizovat -> pak by melo
%byt rozdeleni taky gaussovske nebo normalni varX = sigma^2, sigma=Vrms pro
%gauss rozdeleni
SNR = 20*log10(sqrt(selVPS)/0.707);
figure(1); clf;
SURF(selD,selN,SNR); colormap jet;
if (GAUSS)
title(sprintf('SNR pro signal sin(x), gauss jitter sigma=%0.2g procenta
periody',100*p/(2*pi)));
else
title(sprintf('SNR pro signal sin(x), uniform jitter a=%0.2g procenta
periody',100*p/(2*pi)));
end
xlabel('hloubka bufferu D');
ylabel('pocet vzorku na periodu');
zlabel('SNR [dB]');
```

Příloha 3

```
function [Hj H] = getQuasinHist(gauss,N,p,D,Jinterval,Vinterval)

%gauss - 1 = normalni roznozeni, 0 = rovnomerne
%N - pocet vzorku na periodu
%

%gauss=1;
%a=1;
%sigma=0.5;
%N=350;
%D=1;
%interval=[-2:0.1:2];

t=(2*pi/N):(2*pi/N):(2*pi);

xsin=(2*pi/(N.*(D-1))).*cos(t);

if (gauss)
  sigma=p;
  yrnd=(1/(D-1)).*round((D-1).*normrnd(0,sigma/sqrt(D-1),[3000 1]));
else
  a=p;
  yrnd=(1/(D-1)).*round((D-1).*unifrnd(-(a/sqrt(D-1)),a/sqrt(D-1),[3000 1]));
end;

Hj = histc(yrnd,Jinterval);

qsr = yrnd*xsin;

%figure(2); clf;
foo = size(qsr);
len = foo(1)*foo(2);
quasirnd=reshape(qsr',1, len);
%stairs(quasirnd); grid on;
%figure(3);clf;
H=histc(quasirnd,Vinterval)./len;%grid on;

if (gauss)
%title(sprintf('histogram normalni rozlozeni j(i), pocet vzorku na
periodu=%d',length(t)));
else
%title(sprintf('histogram rovnomerne rozlozeni j(i), pocet vzorku na
periodu=%d',length(t)));
end
```

Příloha 4

viz také annex4.ods (openOffice spreadsheet)

SNR 1 vzorek p=0.2

D	2	3	4	5	10	20	50	100	200
N									
20	-13.3	-22.7	-27.9	-32.0	-42.4	-52.5	-64.6	-73.7	-82.8
30	-13.7	-22.9	-28.2	-32.1	-42.6	-52.2	-64.7	-73.9	-83.1
60	-13.9	-22.9	-28.4	-32.1	-42.6	-52.4	-65.0	-74.1	-83.2
150	-14.5	-23.5	-28.6	-32.6	-43.3	-52.9	-65.2	-74.4	-83.5
200	-14.9	-23.8	-29.1	-32.8	-43.1	-53.1	-65.4	-74.5	-83.8

SNR 2 vzorek p=0.5

D	2	3	4	5	10	20	50	100	200
N									
20	-6.6	-15.7	-21.1	-24.9	-35.5	-45.2	-57.6	-66.8	-75.9
30	-7.2	-16.6	-21.7	-25.4	-36.0	-45.9	-58.0	-67.3	-76.5
60	-8.2	-17.4	-22.6	-26.3	-36.9	-46.8	-58.9	-68.3	-77.4
150	-9.9	-18.9	-24.1	-27.9	-38.4	-48.2	-60.6	-69.7	-78.8
200	-10.4	-19.4	-24.6	-28.4	-39.0	-48.7	-61.1	-70.3	-79.4

SNRE: p=0.2

D	2	3	4	5	10	20	50	100	200
N		9.4	14.6	18.7	29.1	39.2	51.3	60.4	69.5
20		9.2	14.5	18.4	29.0	38.6	51.0	60.2	69.4
30		9.0	14.5	18.3	28.8	38.5	51.1	60.2	69.4
60		9.0	14.1	18.1	28.7	38.4	50.7	59.9	69.0
150		8.9	14.2	17.9	28.2	38.2	50.5	59.6	68.9
200									

SNRE: p=0.5

D	2	3	4	5	10	20	50	100	200
N	D	3	4	5	10	20	50	100	200
20	N=20	9.1	14.5	18.3	28.9	38.6	51.0	60.3	69.3
30	N=30	9.4	14.5	18.2	28.8	38.7	50.8	60.1	69.3
60	N=60	9.1	14.4	18.1	28.6	38.5	50.7	60.0	69.1
150	N=150	9.0	14.2	18.0	28.6	38.3	50.7	59.8	68.9
200	N=200	8.9	14.2	17.9	28.6	38.3	50.7	59.9	69.0

odchylka dB	-0.34	-0.06	-0.4	-0.18	-0.62	-0.29	-0.11	-0.23
	0.16	-0.03	-0.18	-0.15	0.11	-0.22	-0.1	-0.14
	0.14	-0.09	-0.18	-0.17	0.01	-0.41	-0.2	-0.23
	0	0.09	-0.01	-0.18	-0.08	-0.01	-0.13	-0.1
	0.05	0.01	0.02	0.33	0.12	0.13	0.28	0.09