Waveletek 2.

Korszerű matematikai módszerek a geodéziában

2019.10.28.

További témák

- Szűrés waveletekkel (kerék gyorsulások)
- Diszkrét diadikus WT, skálázó függvények
- Sokskálás analízis (MRA)
- Operátor tömörítés waveletekkel
- Globális wavelet spektrum (idő és skála átlagolás, simítás)
- Wavelet kereszt-spektrum, koherencia
- Többváltozós (komplex) folytonos WT

CWT szűrés

Elve:

Ha a CWT-nek **csak egy adott részéből** rekonstruáljuk az idősort, akkor (idő-frekvencia) **szűrést** végzünk az x_n idősoron

Interaktív Jupyter munkafüzet (Python kóddal)

https://nbviewer.jupyter.org/github/gyulat/Wavelet/blob/master/CWT_filter.ipynb

In [29]:

```
pad = 1
dj = 0.125
s0 = 2*dt
j1 = 10/dj
mother = 'MORLET'

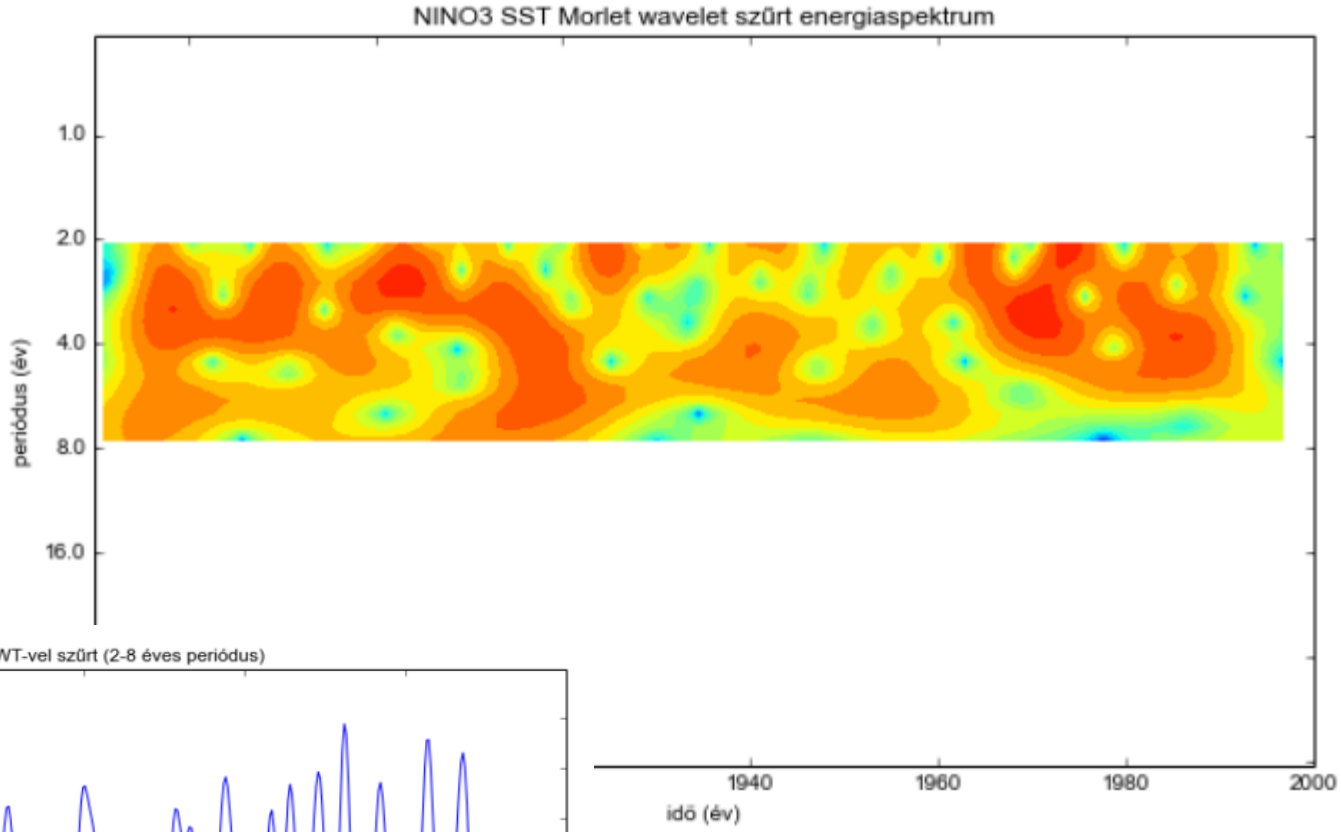
wave, period, scale, coi = wavelet(axi,dt,pad,dj,s0,j1,mother)
# energiaspektrum (Compo)
power = (np.abs(wave)) ** 2
# szűrés - küszöbértékekkel
pvar = np.std(power)**2  # jel variancia
lvl = 0.15     # küszöbszint (jel varianciához viszonyítva)
thr = lvl*pvar
zmask = power <=thr # itt kell kinullázni
powerthr = np.copy(power)
powerthr[zmask] = 0.0
wavethr = np.copy(wave)
wavethr[zmask] = 0.0

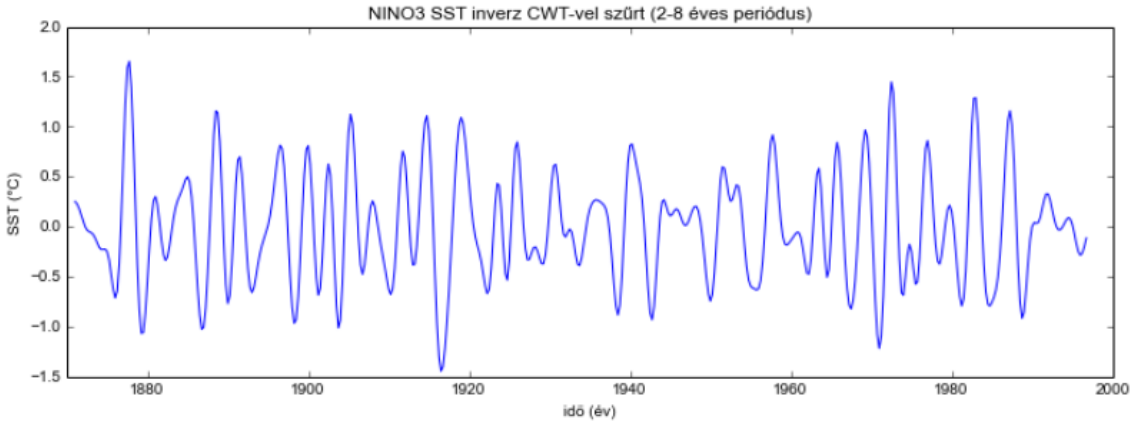
# rekonstrukció inverz wavelet transzformációval
axr = icwt(wavethr, scale, dt, dj, 'morlet')
```

SST szűrés CWT-iCWT-vel

Példa: 2-8
éves periódusú

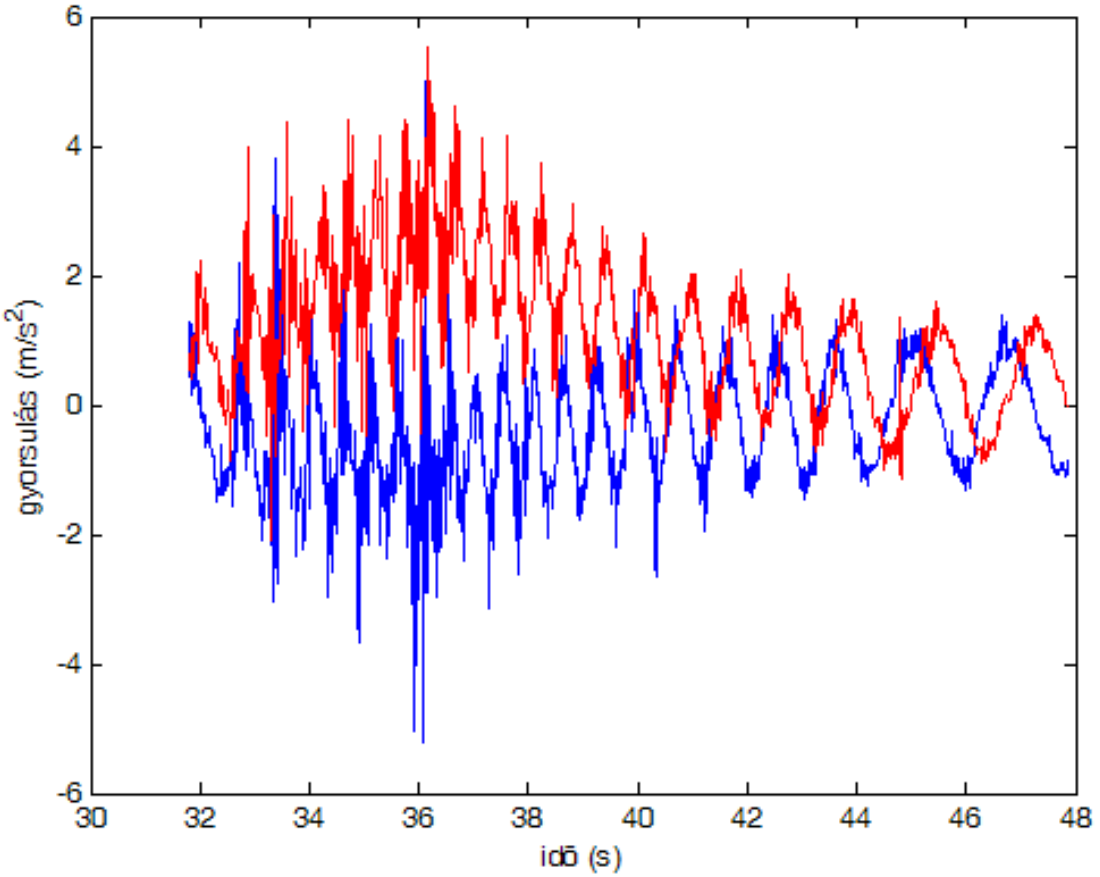
El-Niño SST





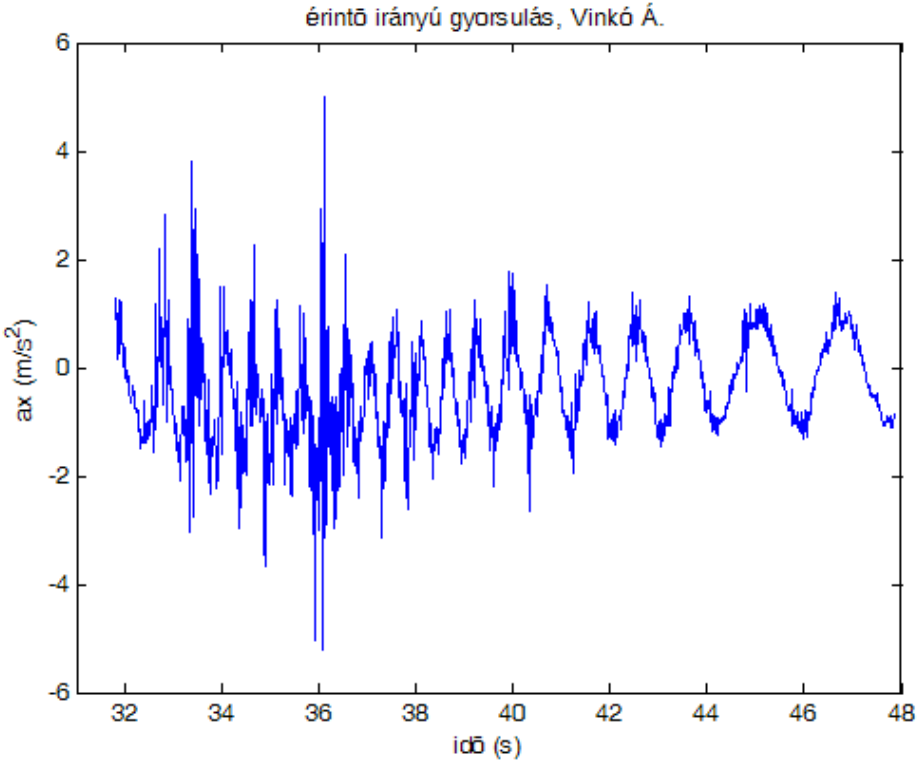
Kerék gyorsulás adatok szűrése waveletekkel

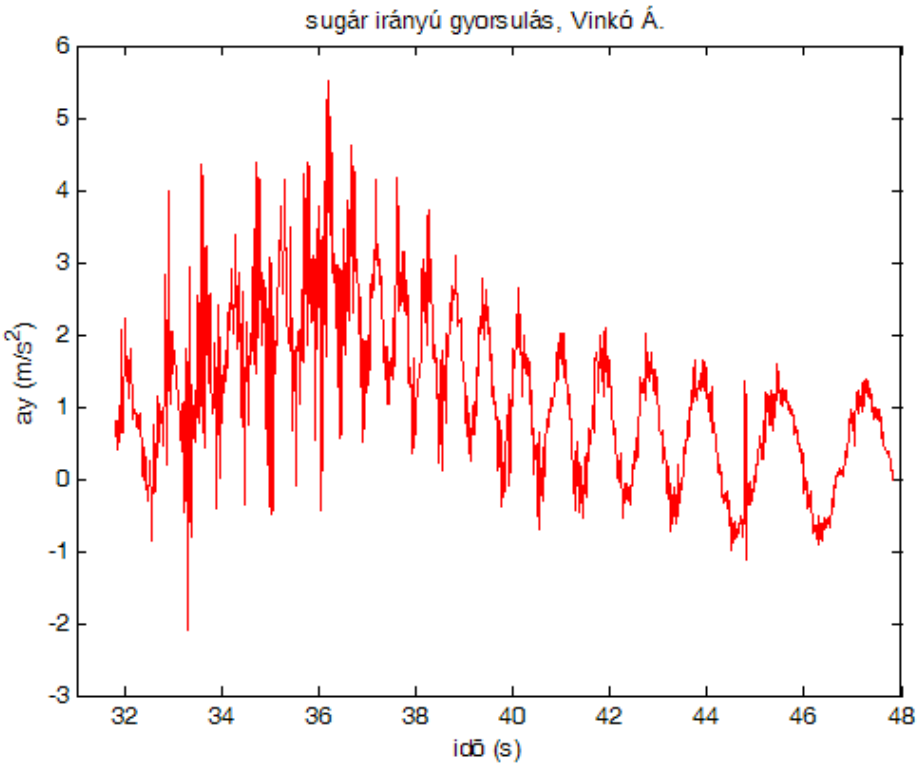
- gyorsulás adatok interpolációja ($\Delta t = 0.01s$)



~10m/s² egység

Érintő és sugár irányú gyorsulás

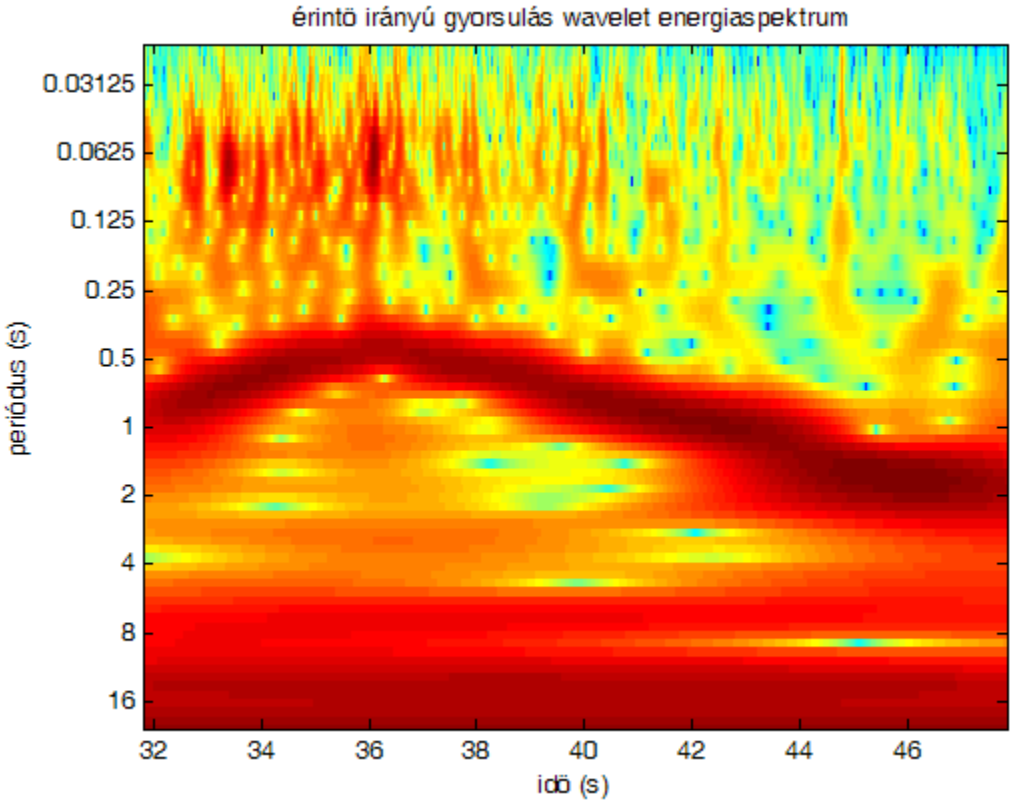


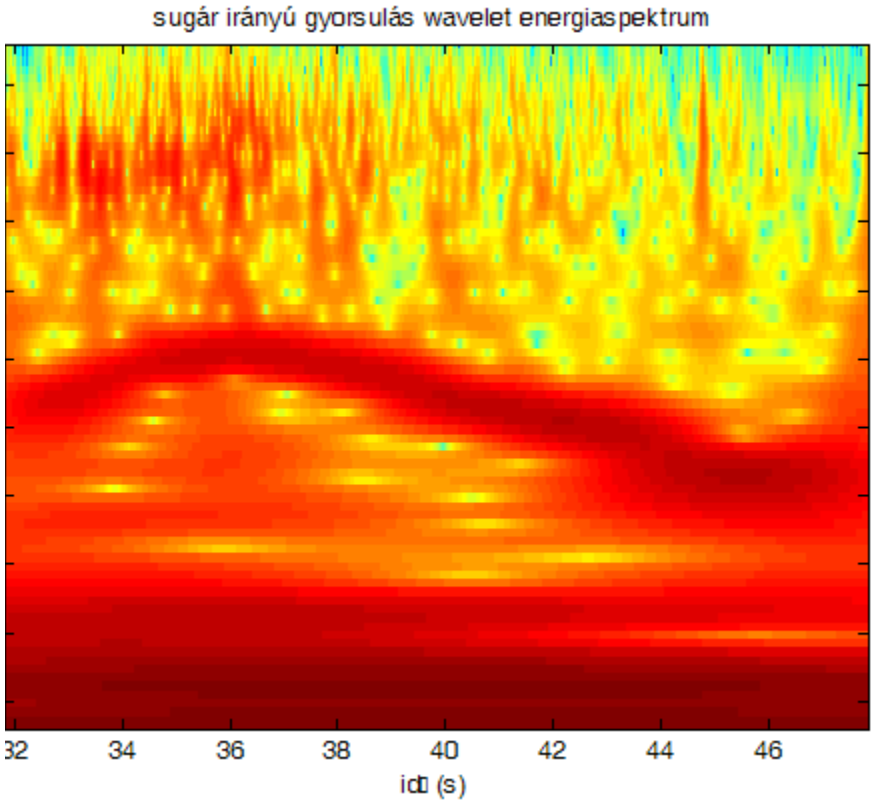


~10m/s^2 egység

Morlet wavelet térképek

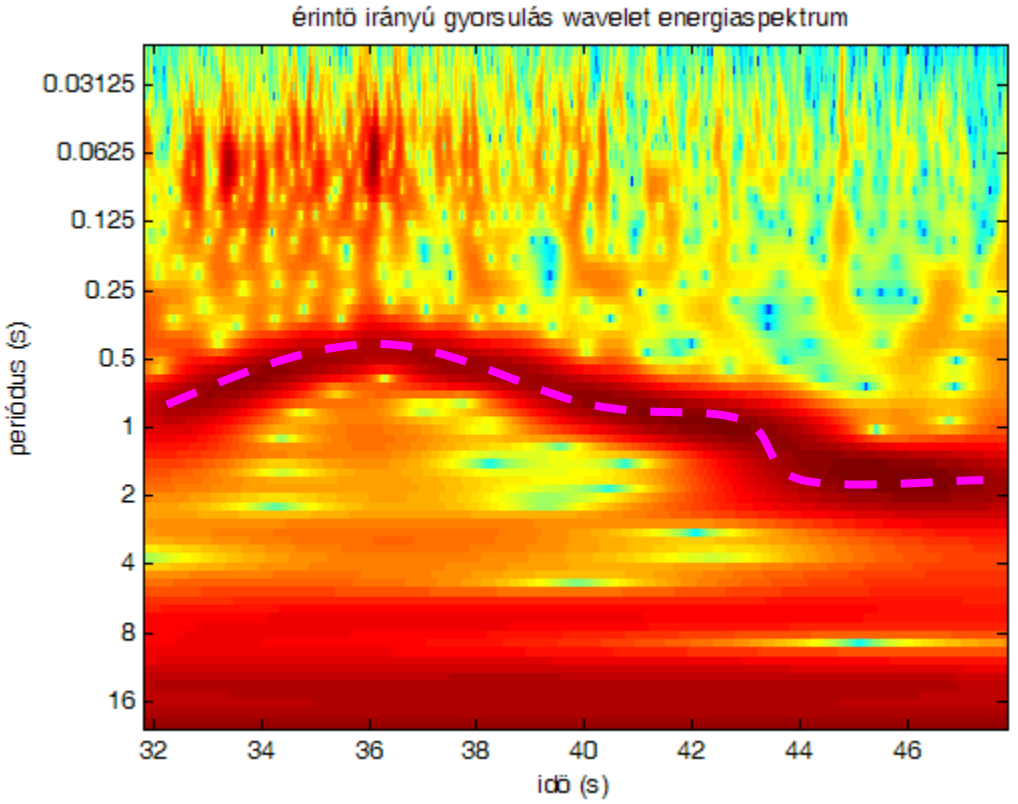
- 10 oktáv, 8 részoktáv ($\Delta t = 0.01s$)

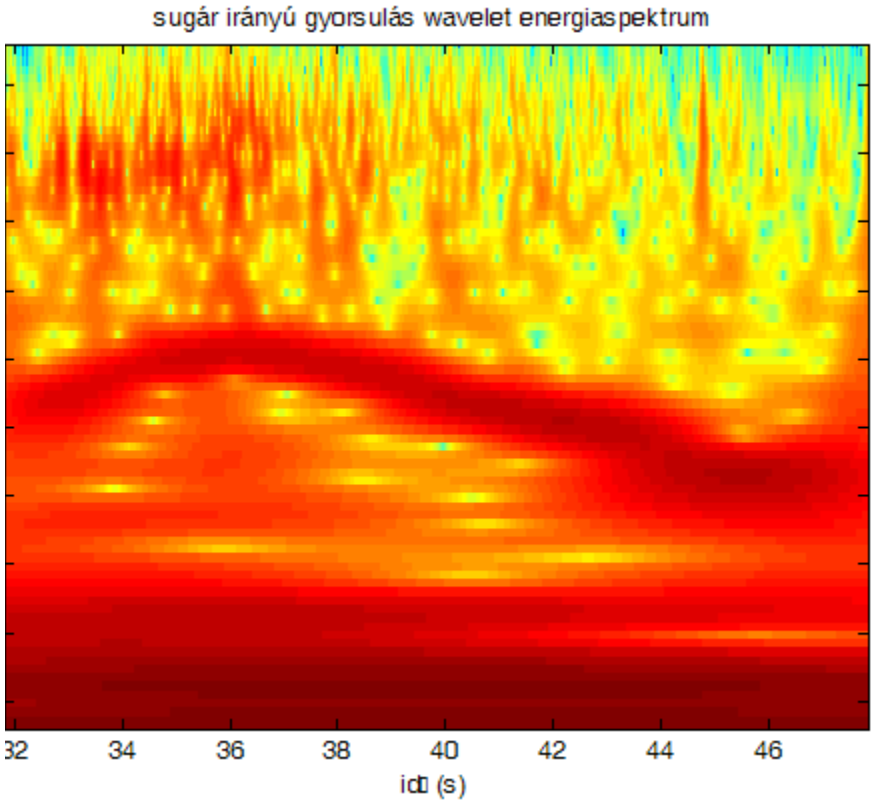




Morlet wavelet térképek

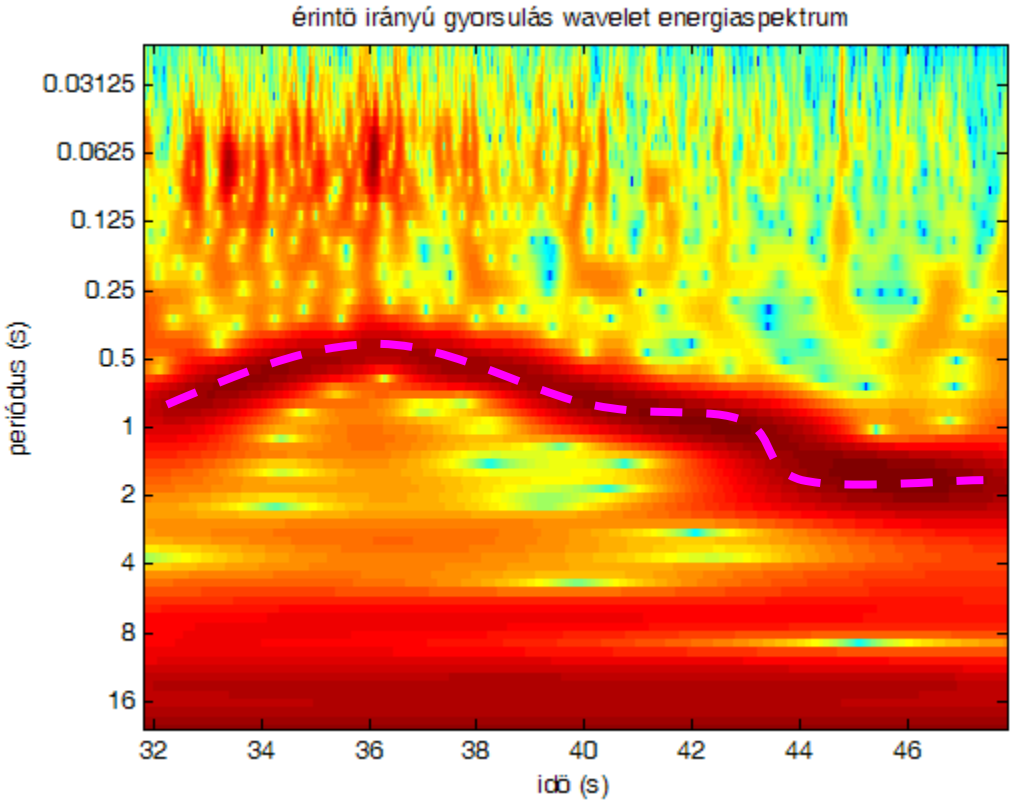
- 10 oktáv, 8 részoktáv ($\Delta t = 0.01s$)

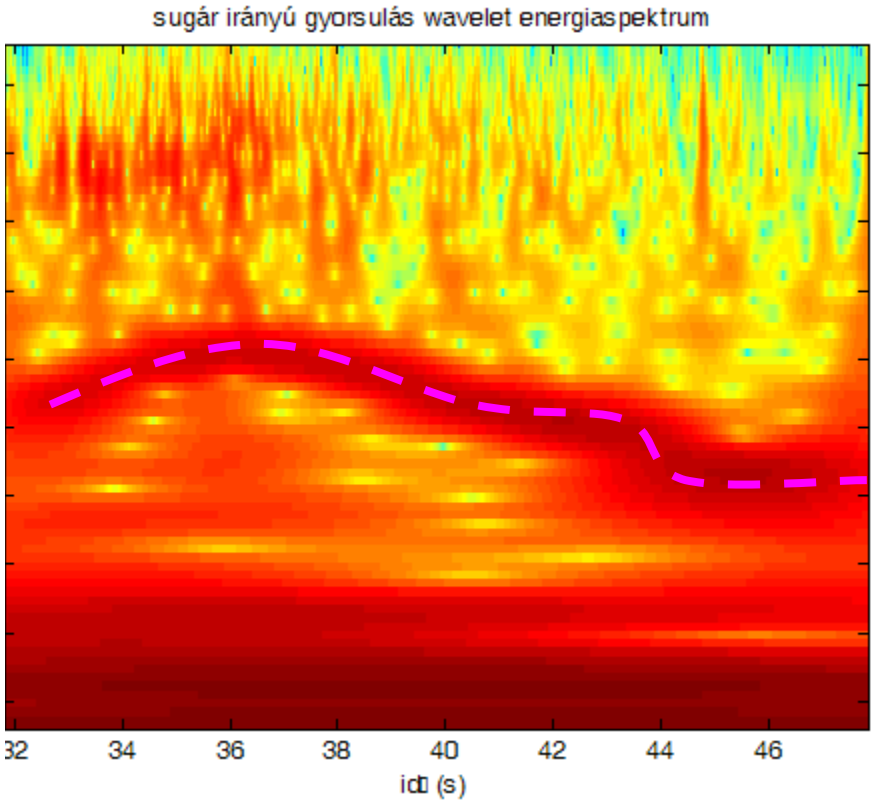




Morlet wavelet térképek

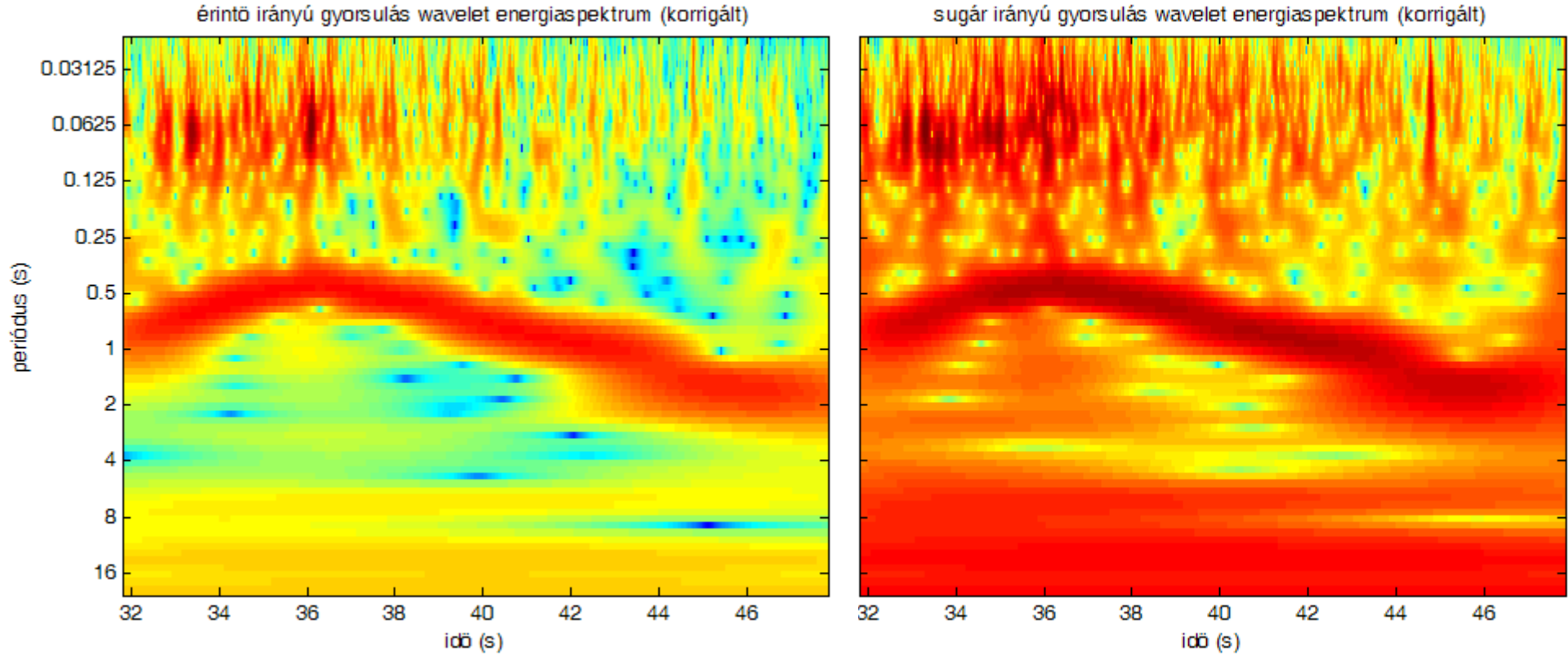
- 10 oktáv, 8 részoktáv ($\Delta t = 0.01s$)





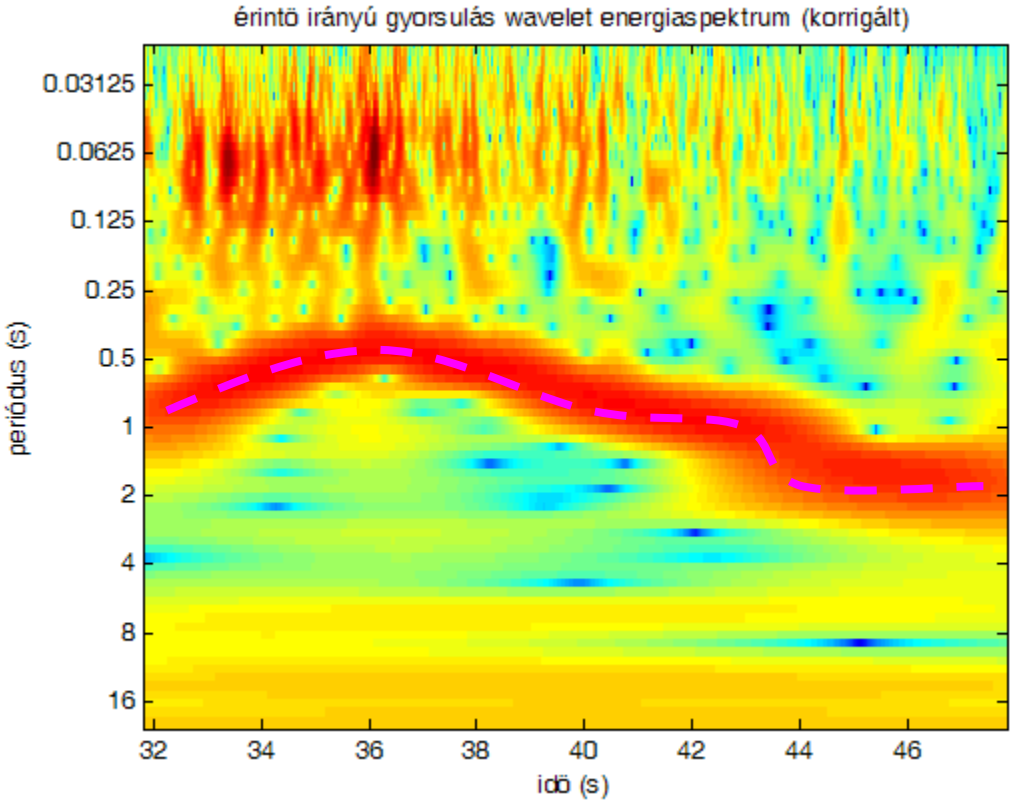
Morlet wavelet térképek

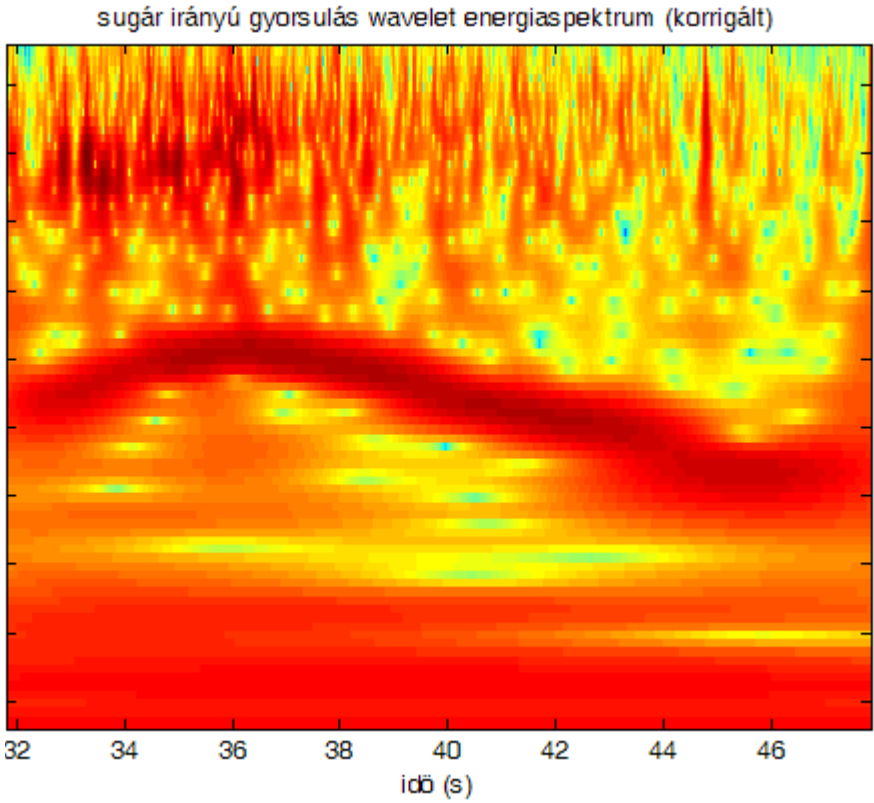
- korrigált teljesítmény normalizációval (Liu et al.)



Morlet wavelet térképek

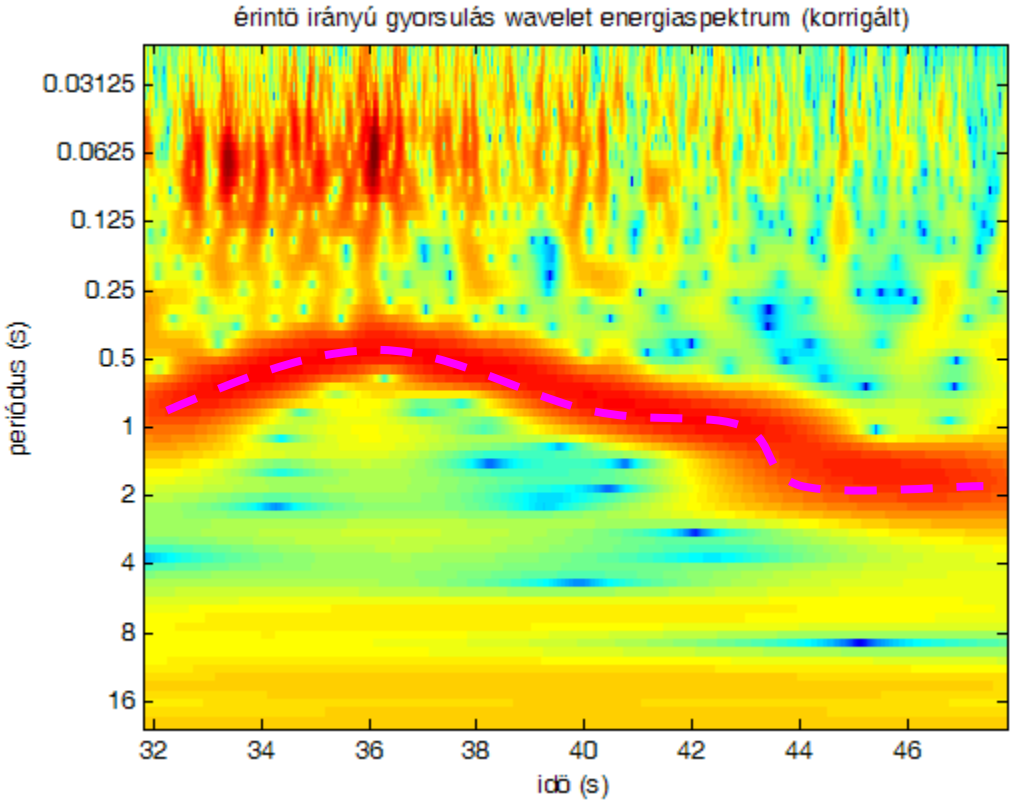
- korrigált teljesítmény normalizációval (Liu et al.)

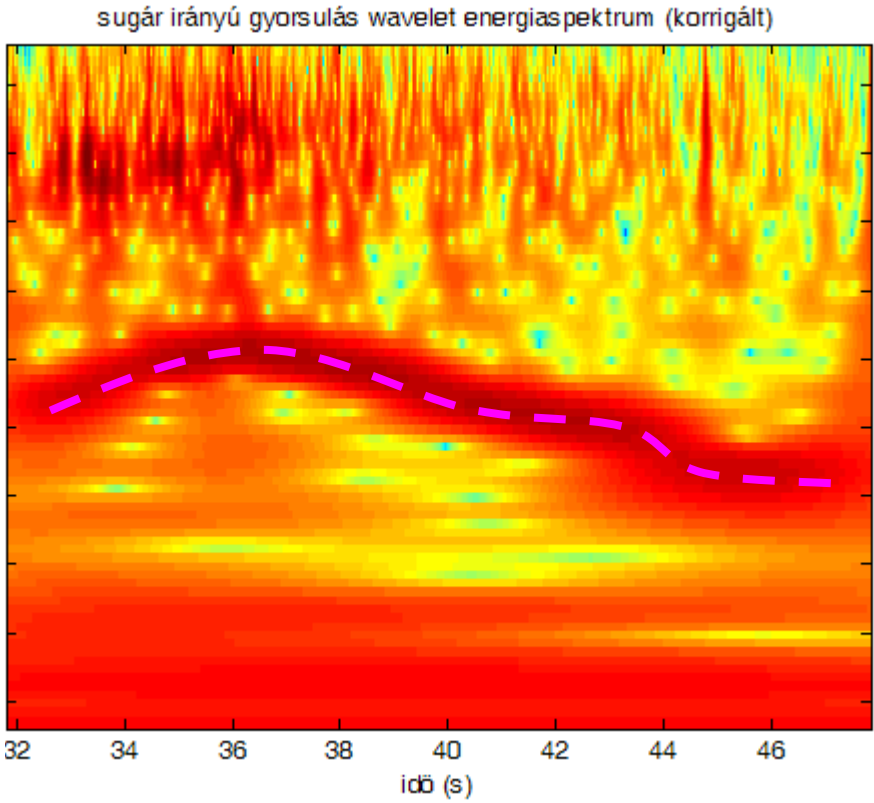




Morlet wavelet térképek

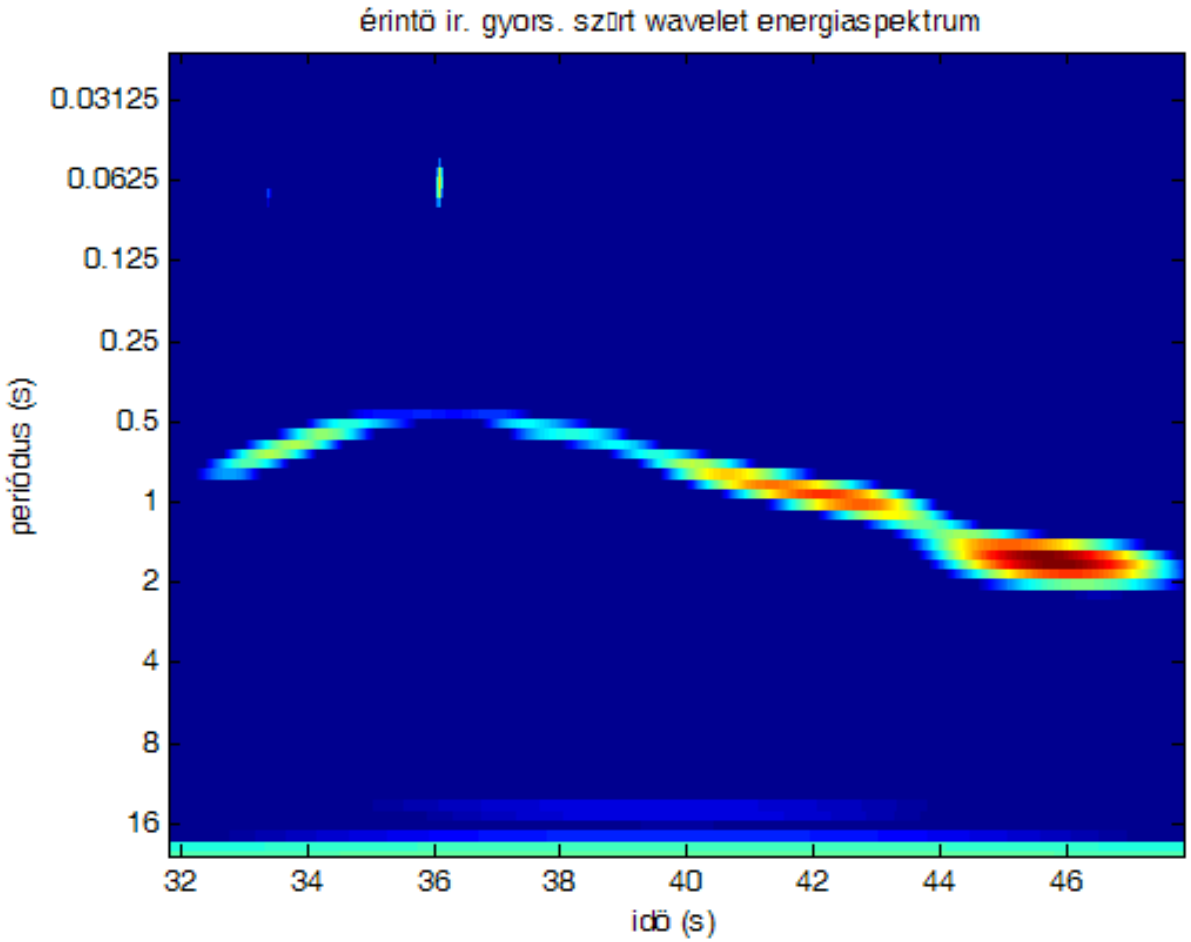
- korrigált teljesítmény normalizációval (Liu et al.)





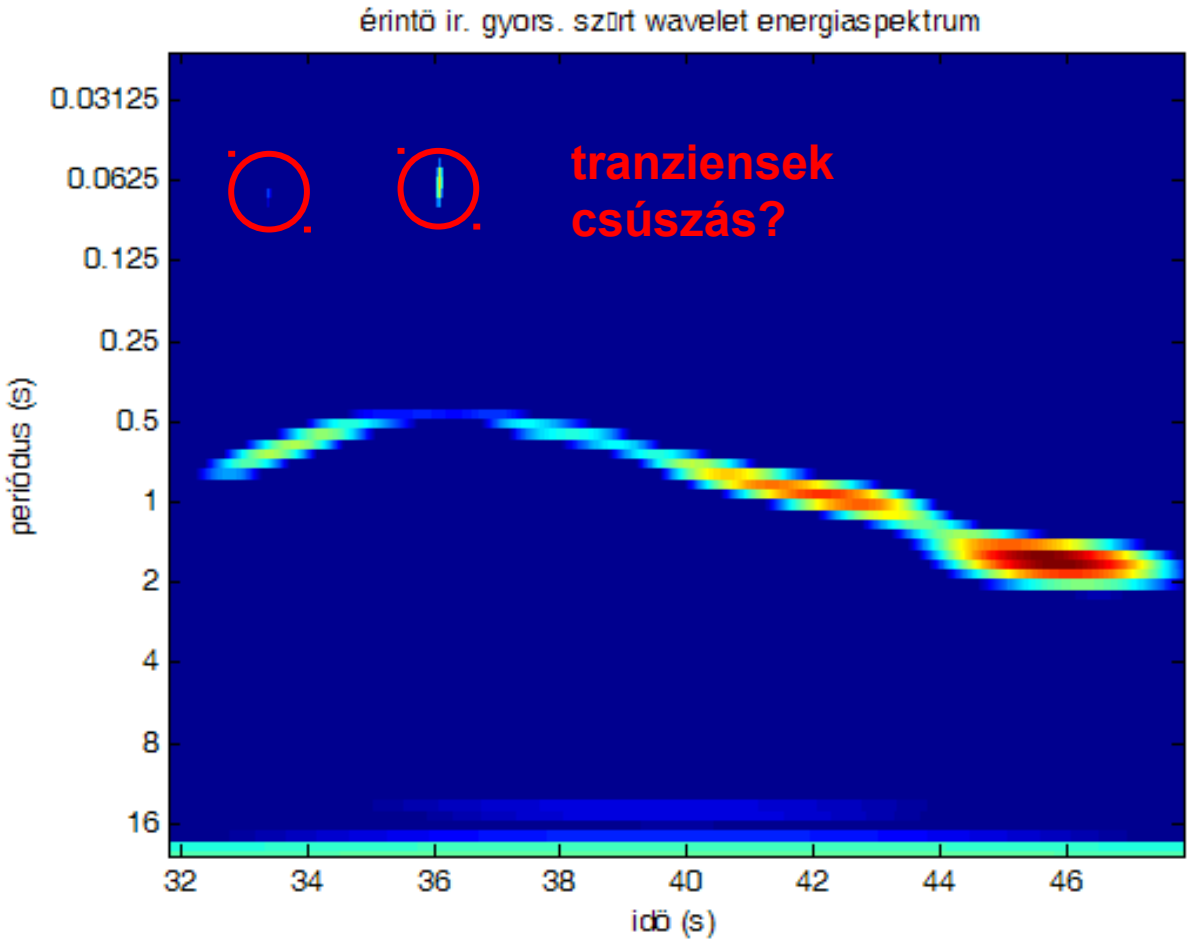
Szűrés

- 10%-os varianciát meghaladó együtthatók



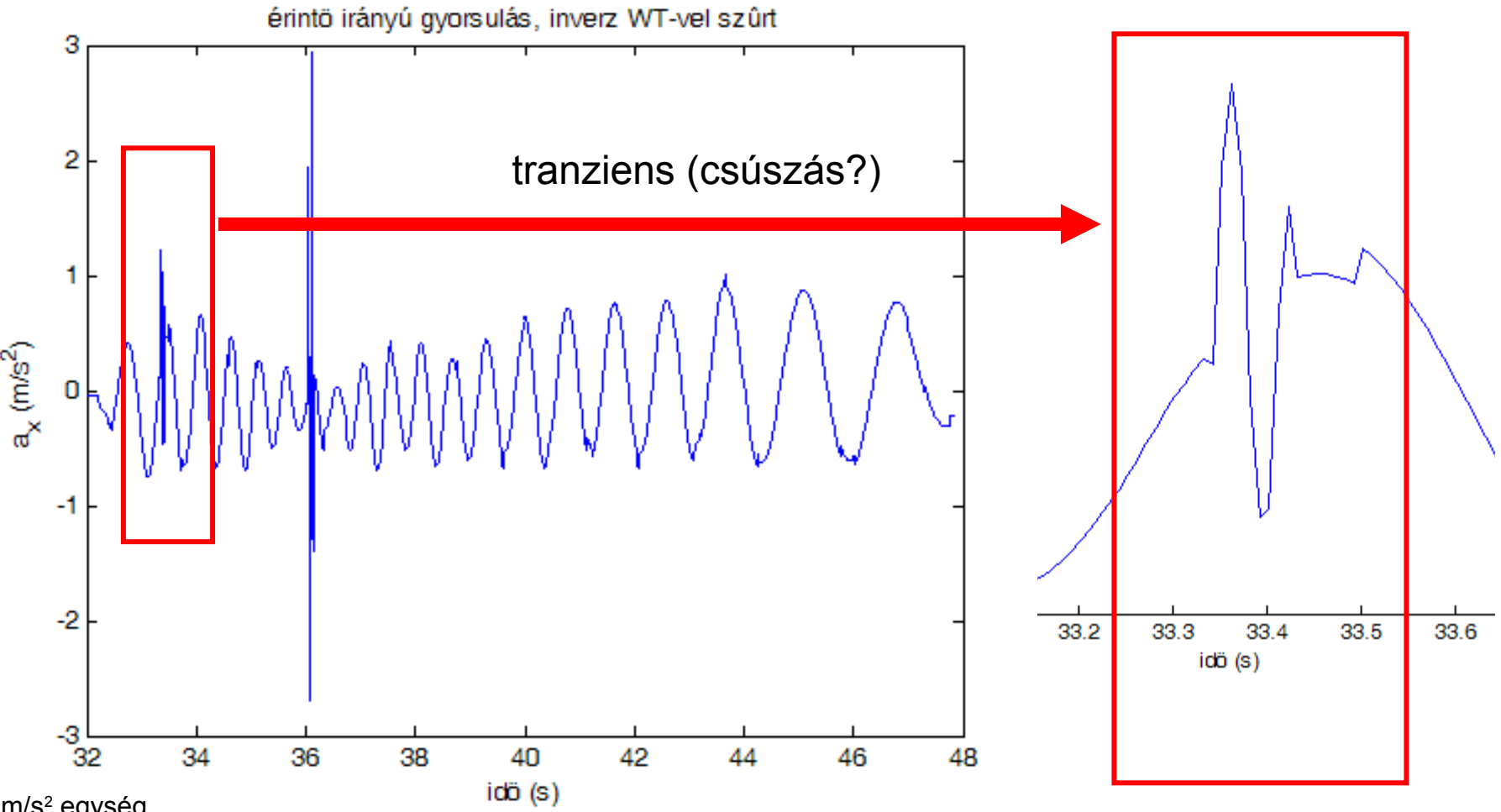
Szűrés

- 10%-os varianciát meghaladó együtthatók



Szűrt érintő irányú gyorsulások

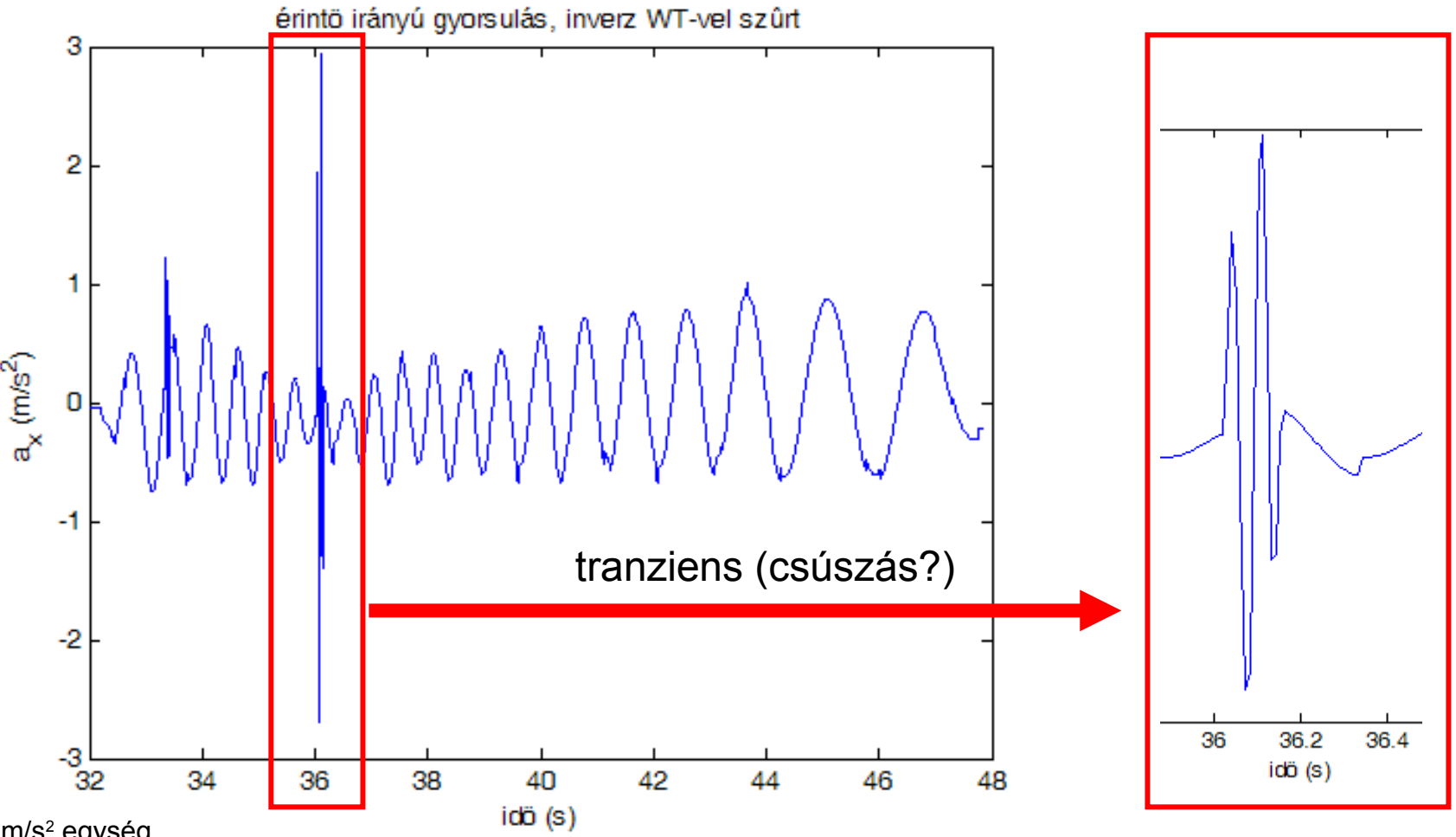
- 10%-os varianciát meghaladó együtthatókkal rekonstruálva



~10m/s^2 egység

Szűrt érintő irányú gyorsulások

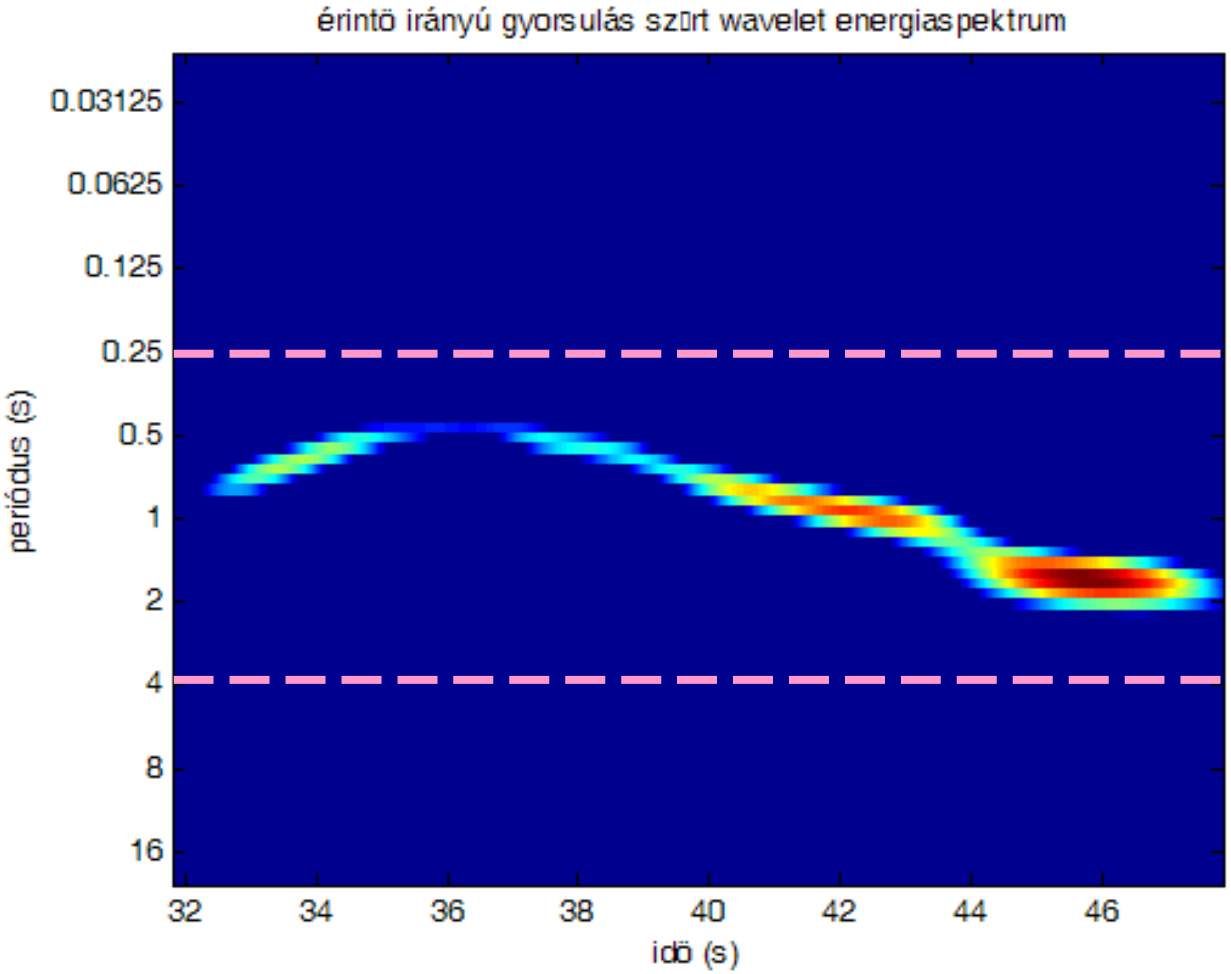
- 10%-os varianciát meghaladó együtthatókkal rekonstruálva



~10m/s^2 egység

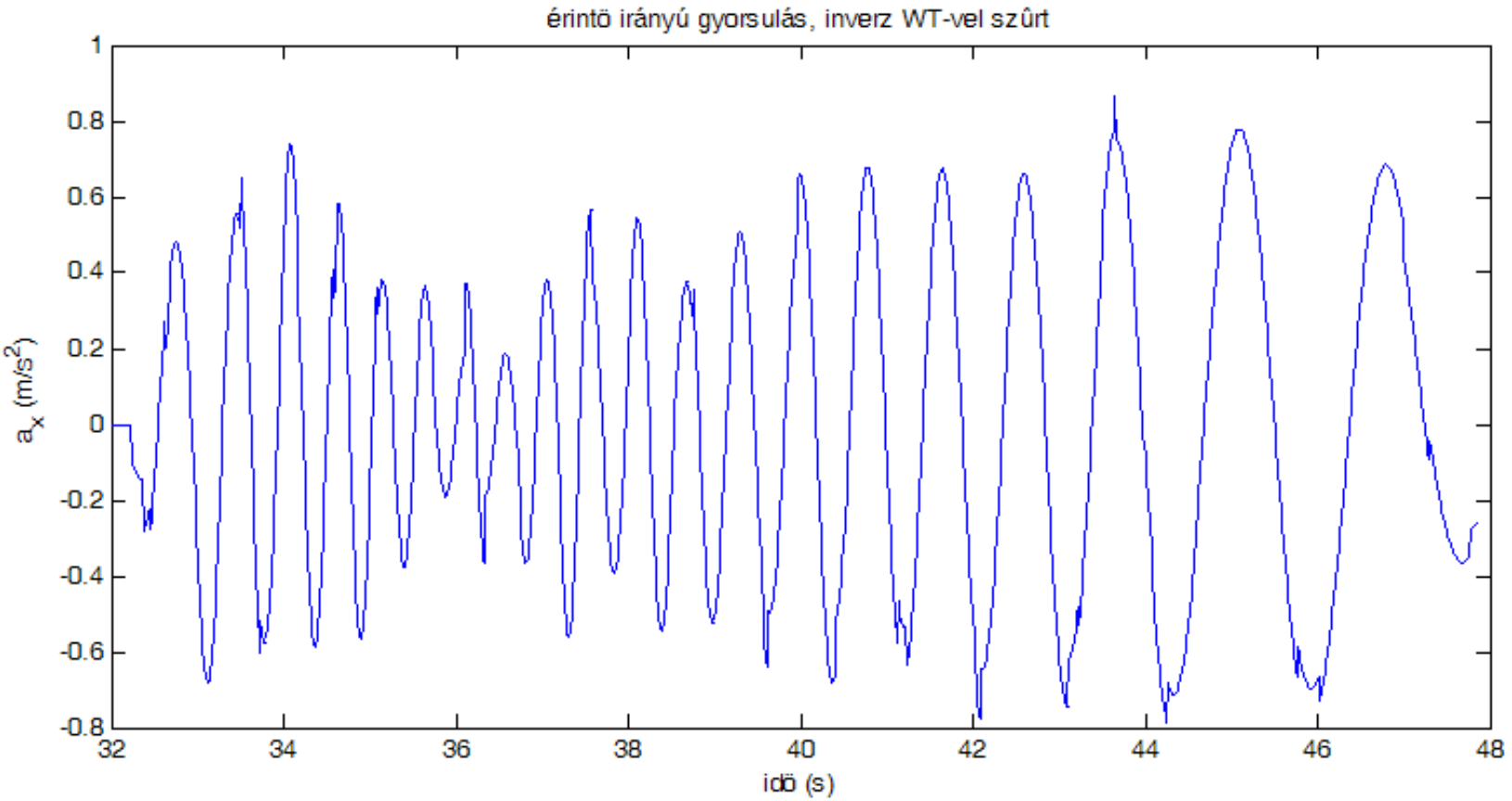
További szűrés

- 0.25 sec – 4 sec közötti periódusú WT együtthatók



Szűrt érintő irányú gyorsulások

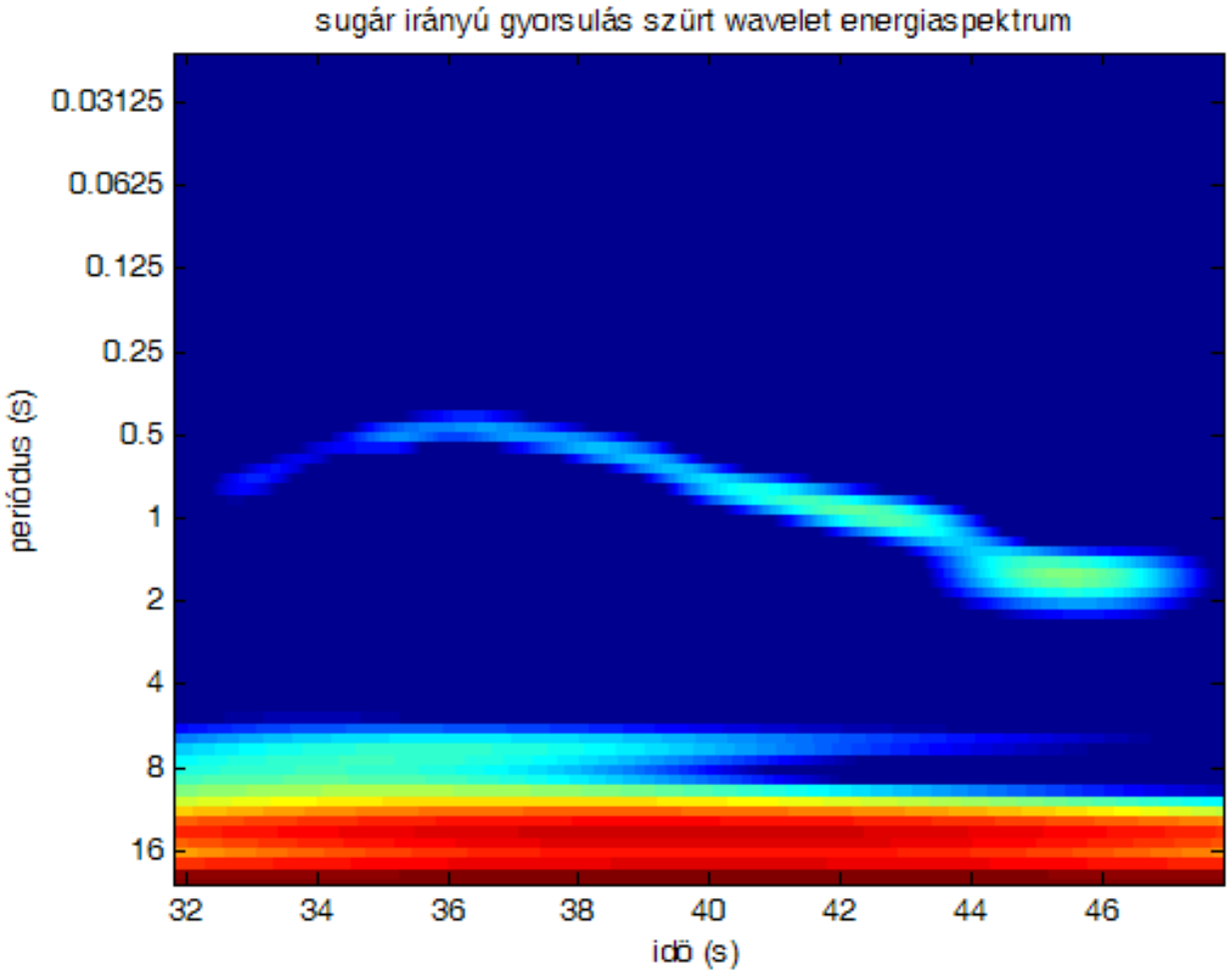
- 0.25 sec – 4 sec közötti periódusú és 10% varianciát meghaladó WT együtthatókkal



~10m/s^2 egység

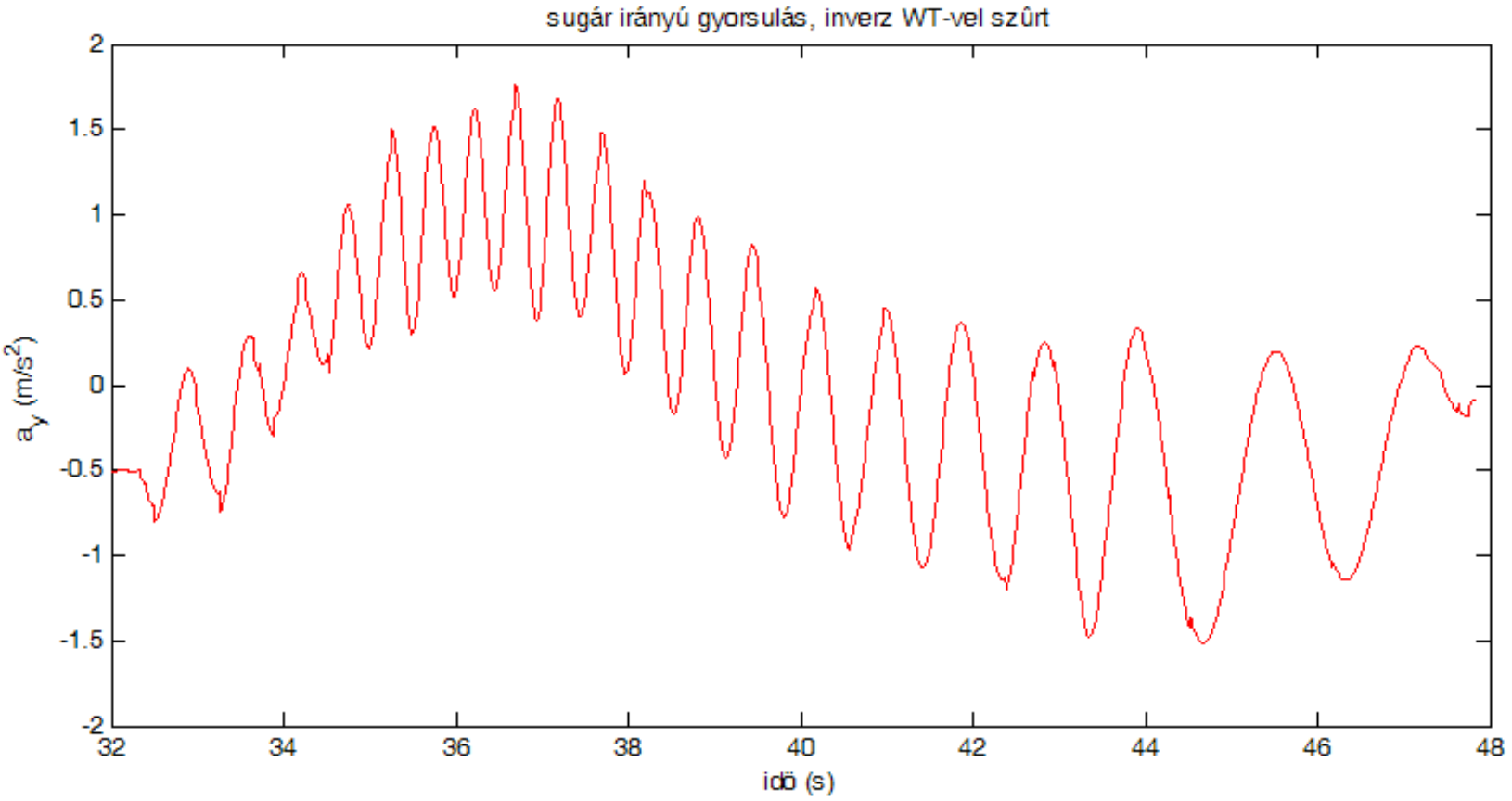
Szűrés – sugár irányú gyorsulások

- 0.4%-os varianciát meghaladó együtthatók



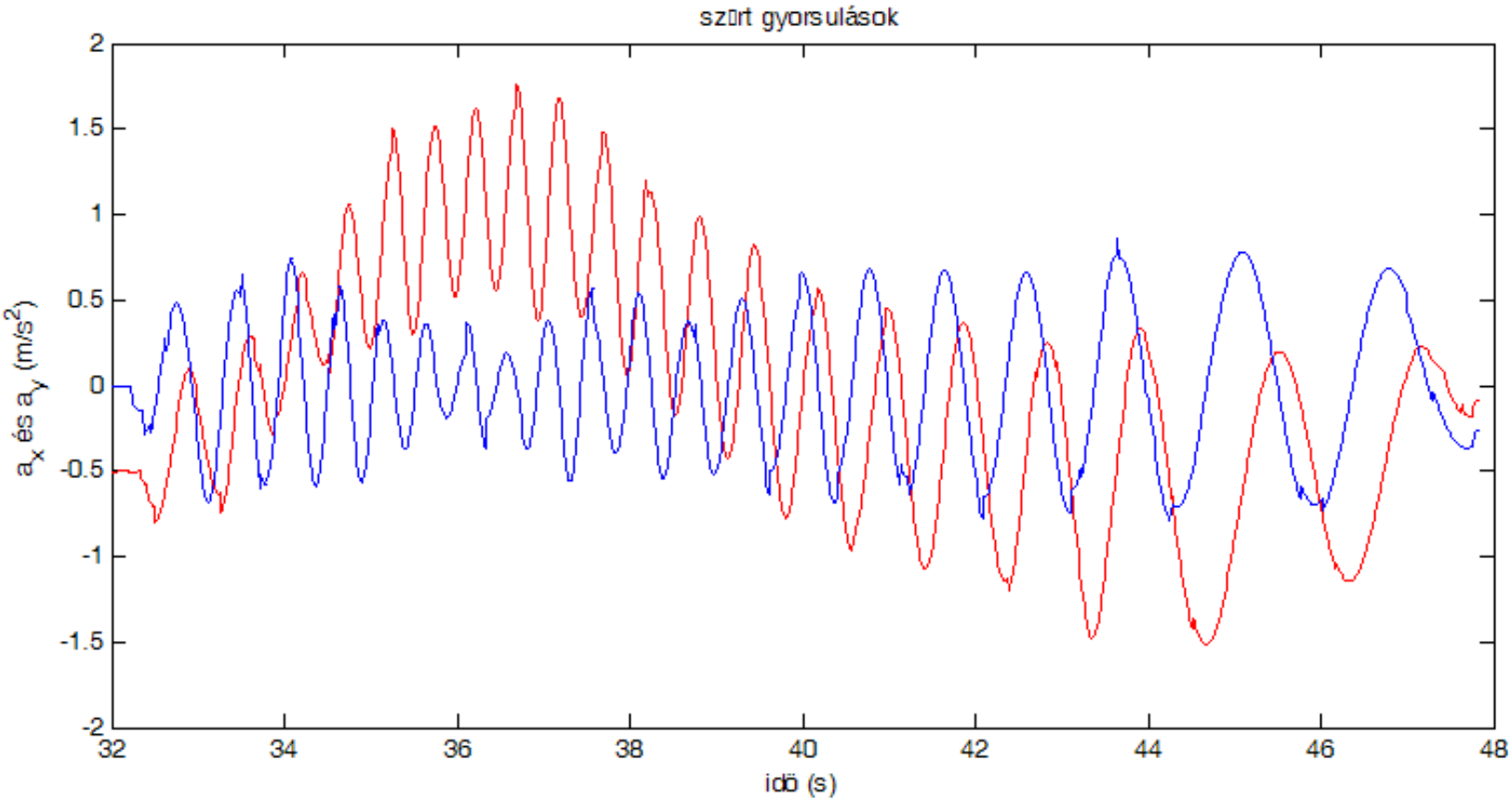
Szűrt sugár irányú gyorsulások

- 0.4%-os varianciát meghaladó együtthatókkal:



~10m/s^2 egység

Szűrt érintő és sugár irányú gyorsulások



~10m/s^2 egység

Zajszűrés inverz CWT-vel

chirp (folyamatosan változó frekvenciájú jel) zajszűrése

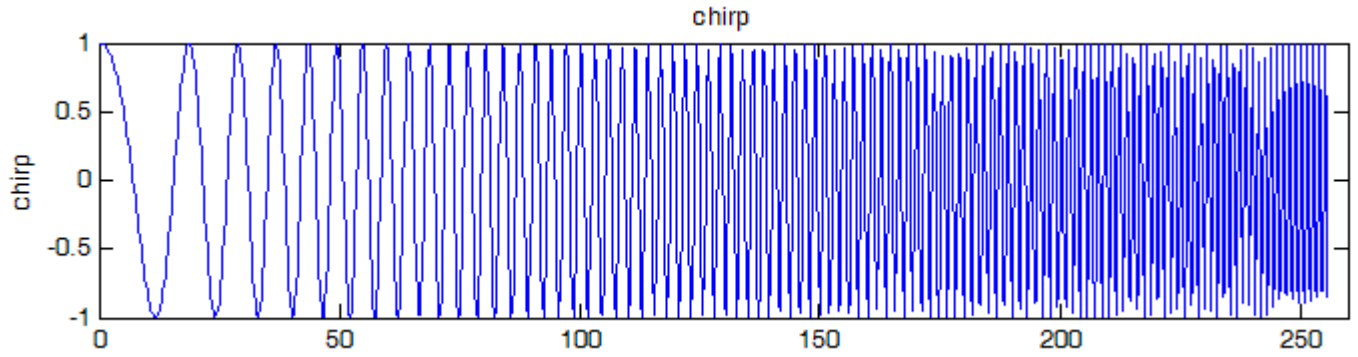
chirp_filter.m

```
N = 1024;
dt = 0.25;
t = dt*[0:N-1];
f1 = 2*pi/4.0;
f2 = 2*pi/256.0;
ch = chirp(t,f2,500,f1);
```

t : időpontok (**dt** : mintavételi időköz)

f2 : 0 időpontbeli frekvencia

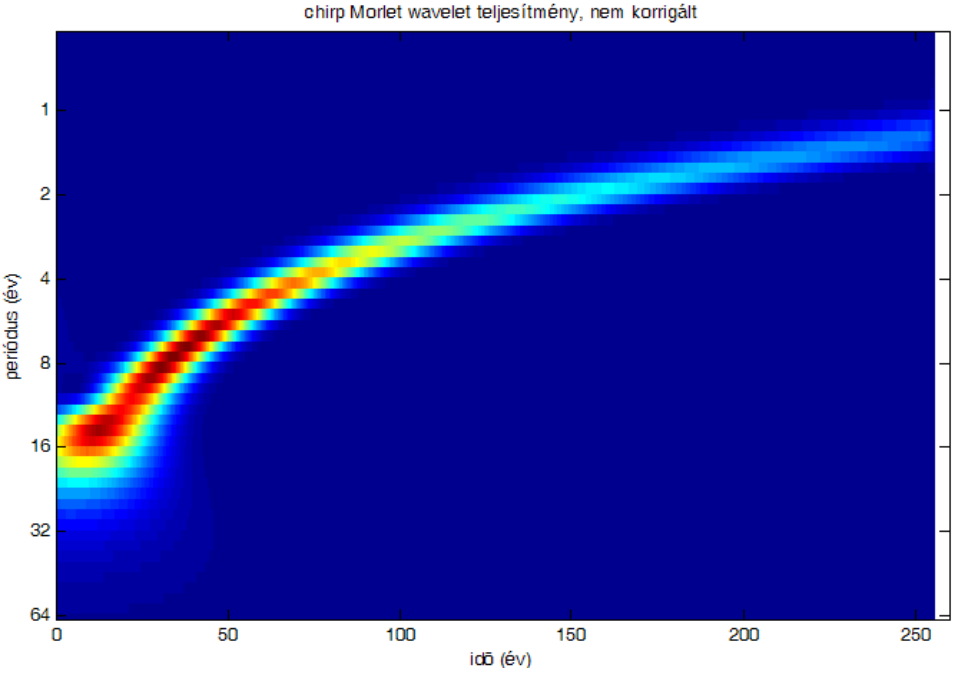
f1 : t = 500 időpontbeli frekvencia

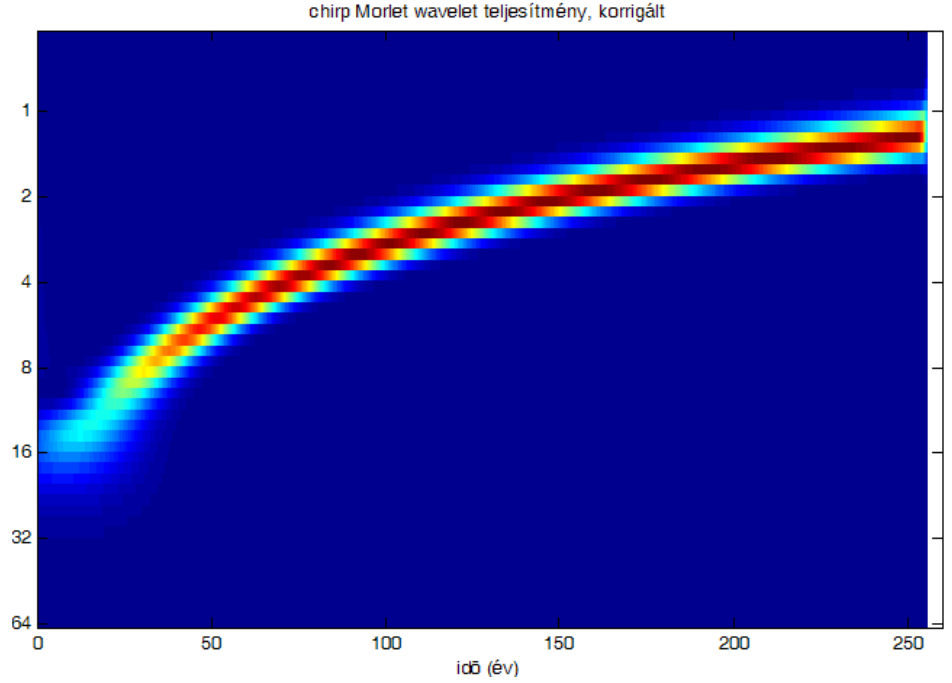


Zajmentes chirp CWT spektrum

chirp_power.m

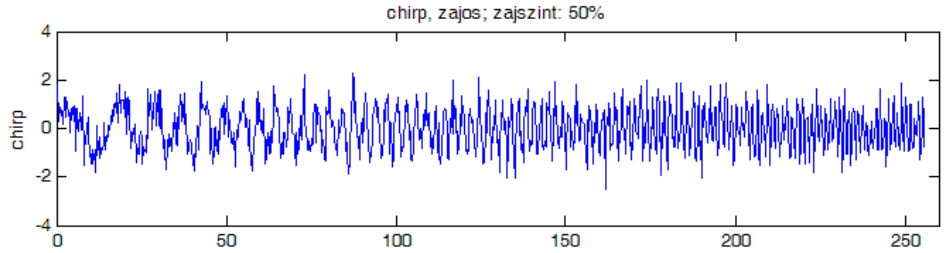
```
[wave,period,scale,coi] = wavelet(ch,dt,pad,dj,s0,j1,mother);
%power = (abs(wave)).^2;      % teljesítmény
scale_ext = (scale')*(ones(1,N));
power = (abs(wave)).^2./scale_ext;  % teljesítmény (korrigált)
```

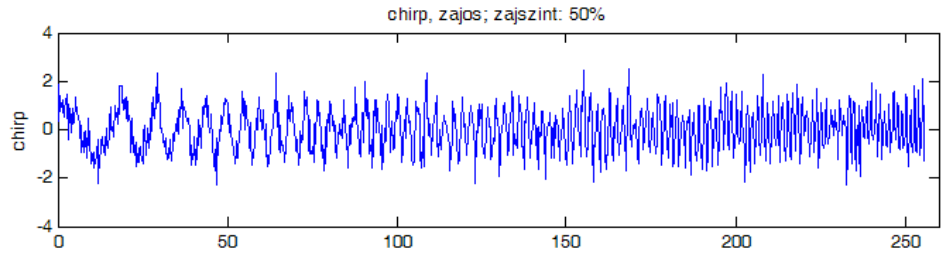


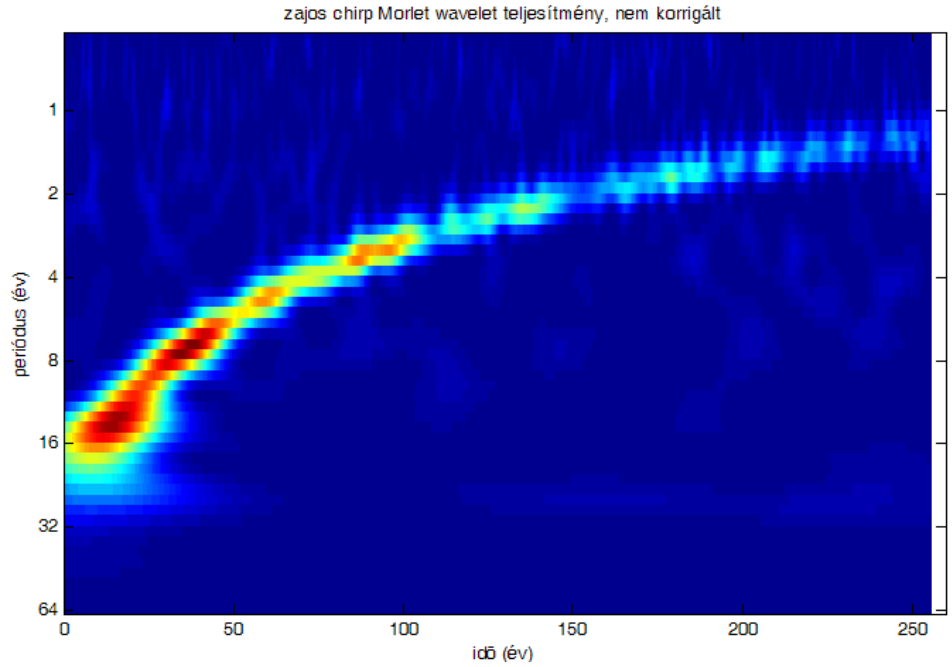


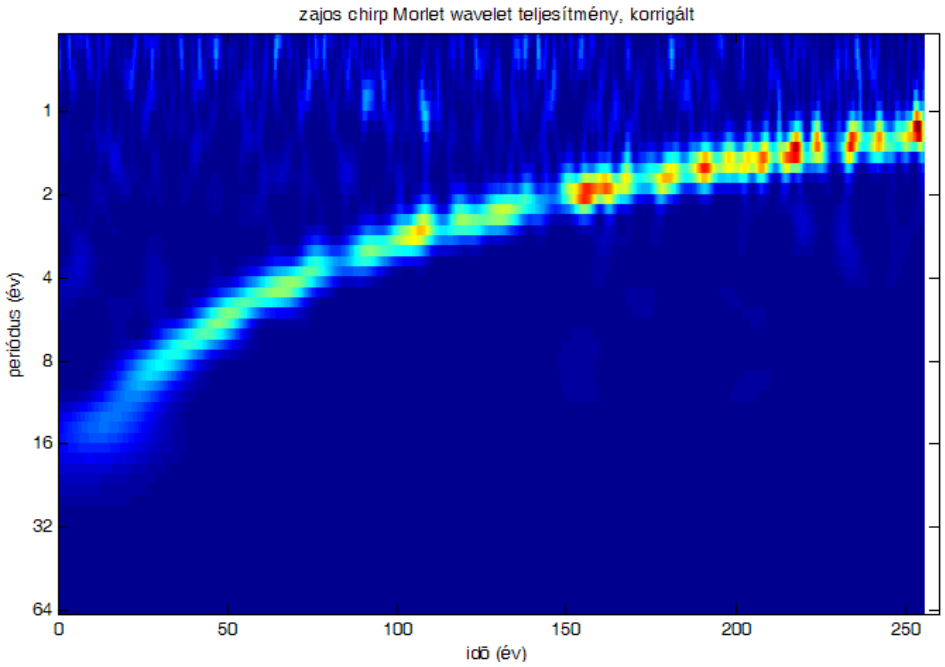
Zajos chirp CWT spektrum

```
nl = 0.5;
noise = nl*randn(1,N);  % Gauss-f. fehérzaj
ch = ch + noise;
```





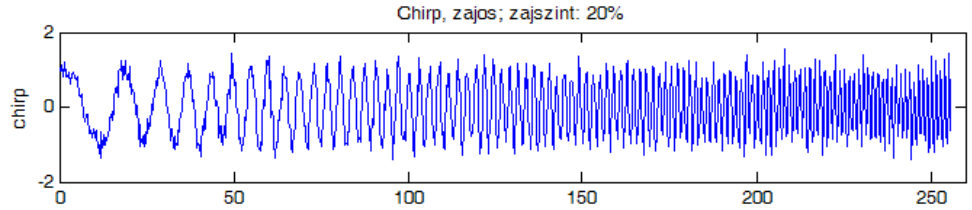


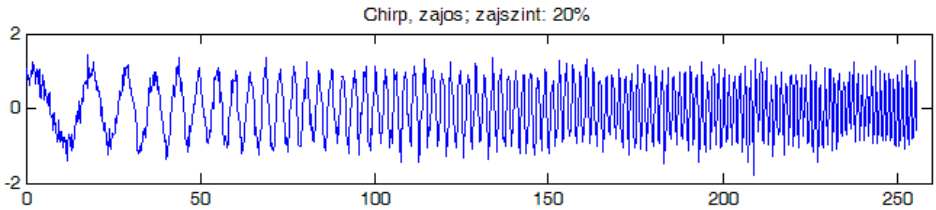


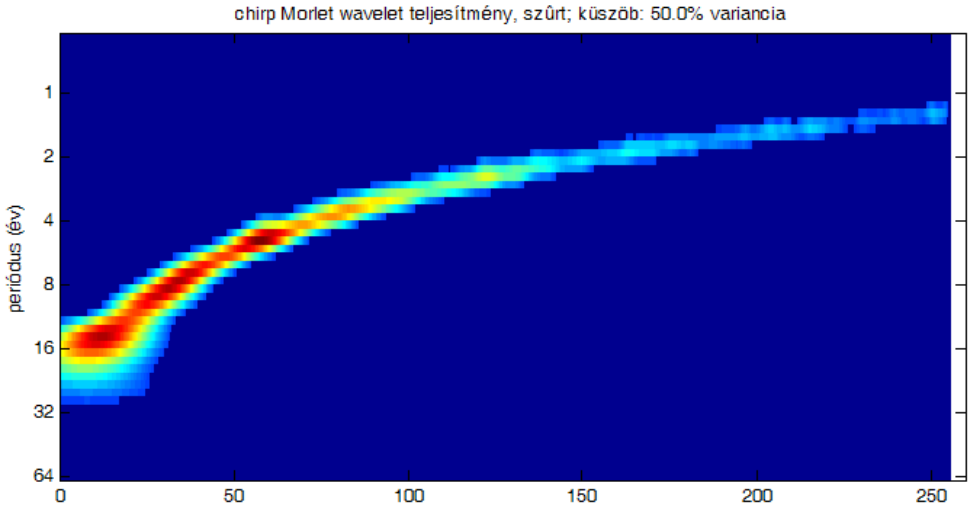
Chirp zajszűrés inverz CWT-vel

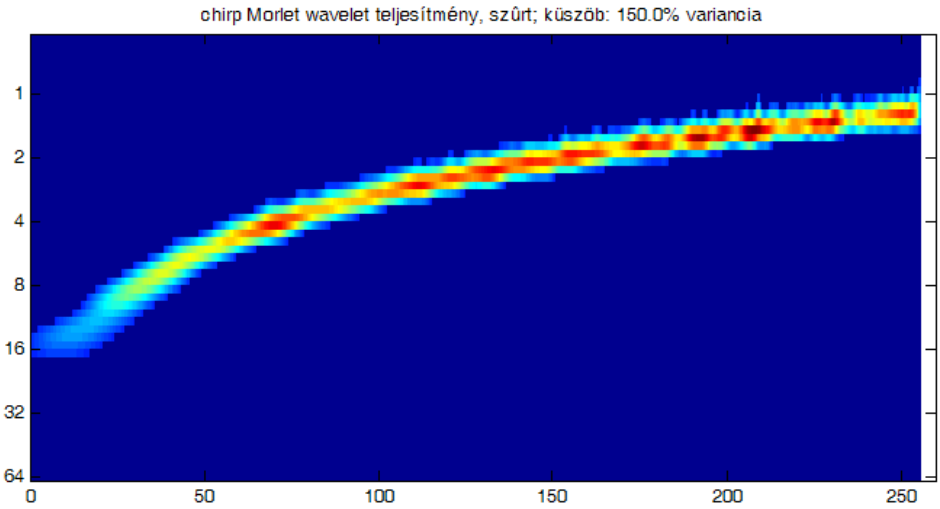
chirp_filter.m

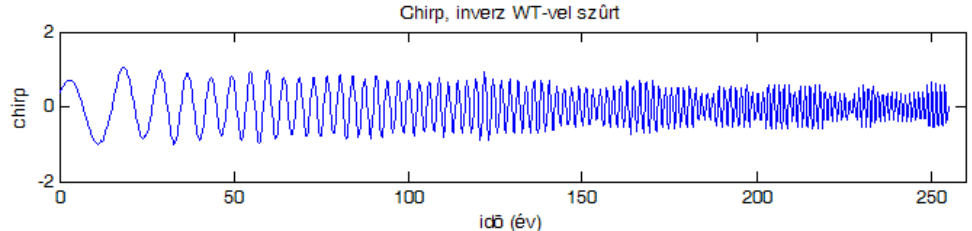
```
% szűrés
pvar = (std(power(:)))^2;
lvl = 0.5;      % küszöbszint
thr = lvl*pvar;
mask = power>thr;
power = power .* mask;
wavethr = mask .* wave;
```

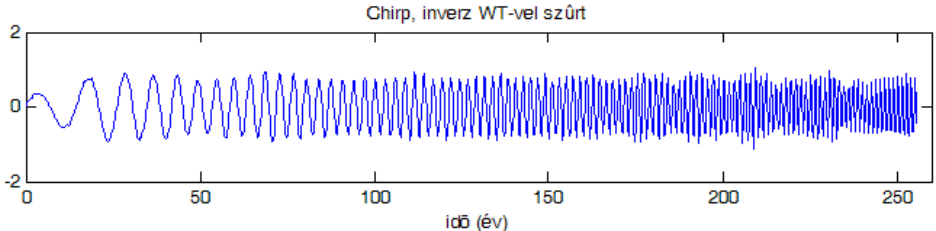






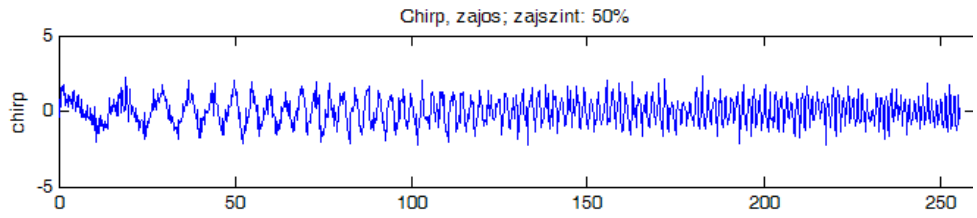


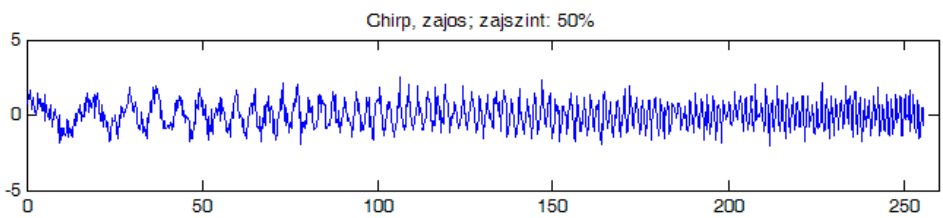


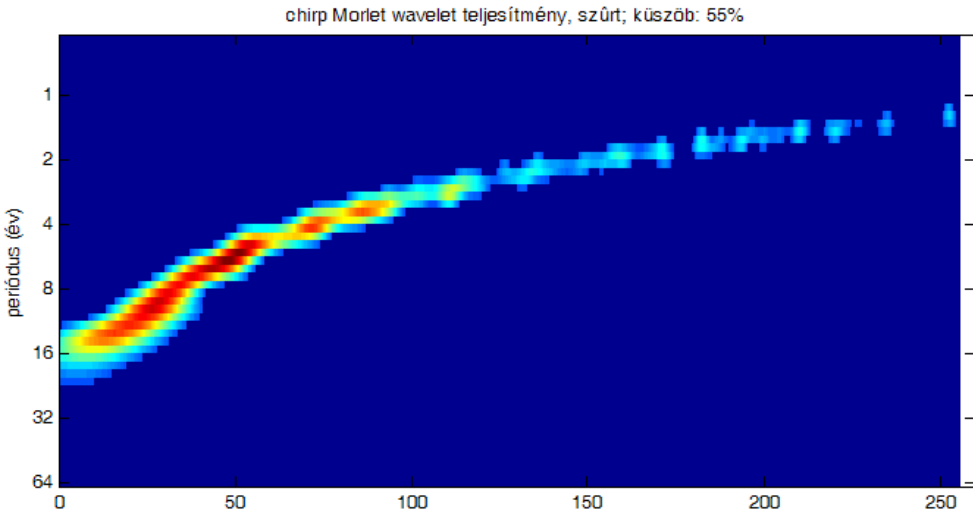


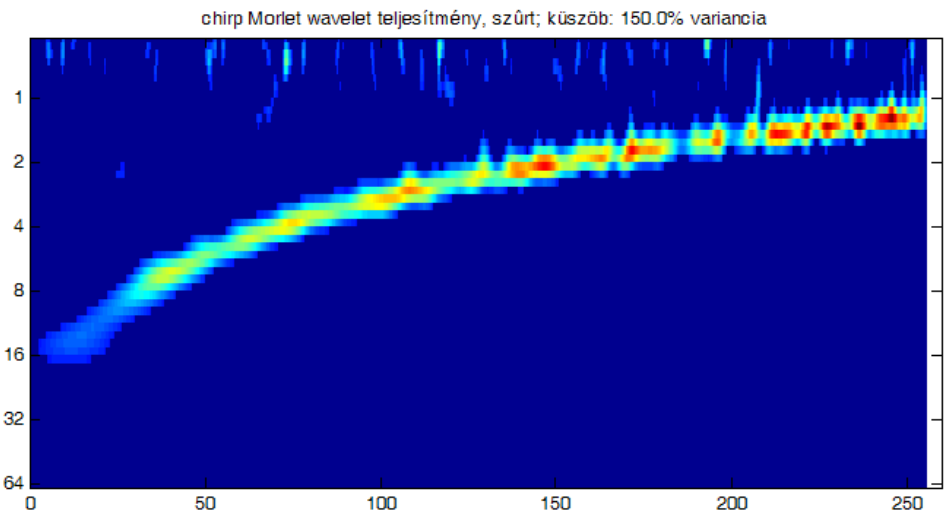
Chirp zajszűrés inverz CWT-vel

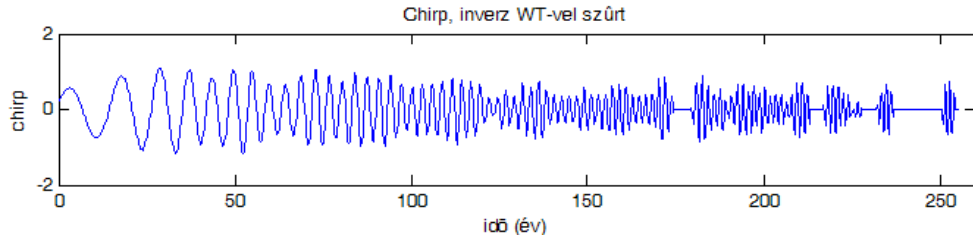
chirp_filter.m (50% fehérzaj)

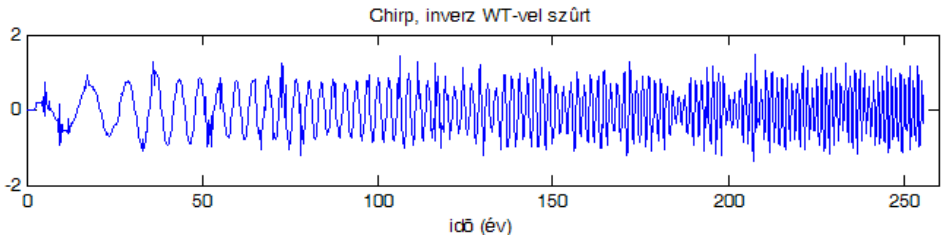












További témák

- diszkrét wavelet transzformáció (DWT)
- kettős tükörszűrők (QMF)
- sokskálás felbontás (MRA)
- operátor tömörítés

CWT és DWT kapcsolata

- Folytonos wavelet transzformáció (CWT)
  - Történetileg korábbi mint a DWT
  - Jel analízishez hasznos
- Diszkrét wavelet transzformáció (DWT)
  - Szűrőcsoportok, tükörszűrők
  - Sokskálás analízis (MRA)
  - Jel feldolgozáshoz (rekonstrukció, szintézis) hasznos
- Sokskálás felbontás (MRA, multiresolution analysis)

Ortogonális diszkrétizáció

CWT: számítjuk az $f(t)$ függvény $\psi_{ab}(t)$ leány waveletekkel (b: eltolás, a: skála) vett konvolúcióját:

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \qquad W_f(a,\, b) = f * \psi_a$$

Mikor lehet az f jelet tökéletesen visszaállítani a mintavételezett folytonos wavelet spektrumból? Legyen a mintavételezés diadikus:

$$a = 2^m, \quad b = n \cdot 2^m, \quad m,\, n \in \mathbb{Z}$$

Diszkrét wavelet felbontás

Az $f(t)$ függvény $\psi_{mn}(t)$ bázisfüggvényekkel (n: eltolás, m: skála) történő felbontása,

$$f(t) = \sum\sum c_{mn}\psi_{mn}(t)$$

$$\psi_{mn}(t) = 2^{-m/2}\psi(2^{-m}t - n)$$

esetében Coifman és Meyer (1986) szerint létezik olyan teljes ortonormált $\psi_{mn}(t)$ bázis, melyre

$$c_{mn} = \int f(t)\psi_{mn}(t)dt = < f,\ \psi_{mn} >$$

Hogyan konstruálhatók alkalmas ψ_{mn} bázisfüggvények?

A választ a sokskálás felbontás (MRA) adja meg:

Ha a $\psi(t)$ wavelet függvény beilleszthető egy MRA-ba, **akkor** a diadikus diszkretizációja ($a = 2^{-j}$, b = $2^{-j}\,k$) elvezet a diszkrét wavelet transzformációhoz (DWT)

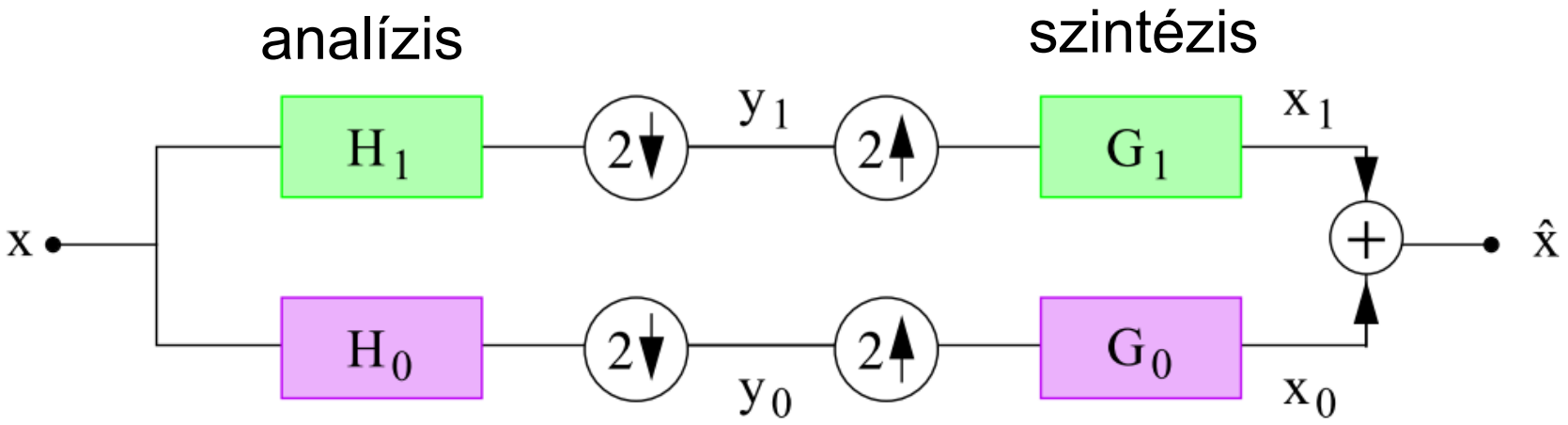
Ha viszont nincsen ilyen MRA az adott $\psi(t)$ wavelethez, akkor ez a diszkretizáció nem ad lehetőséget egyszerű rekonstrukcióra.

Az MRA szoros kapcsolatban van a szűrőcsoportokkal

Kettős tükörszűrők (QMF)

Croisier, Esteban és Galand (1976): Lehetséges egy kettős szűrővel szétosztott és lemintavételezett jelet tökéletesen visszaállítani még akkor is, ha a szűrők nem ideálisak.

$$G_0H_0 + G_1H_1 = I$$

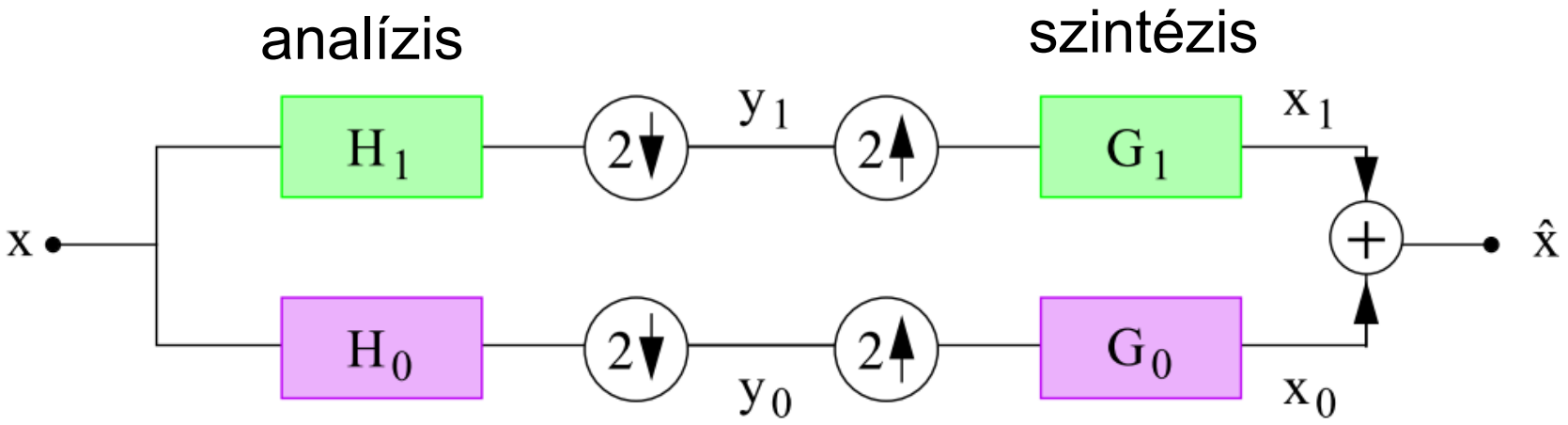


Kettős tükörszűrők (QMF)

Quadrature Mirror Filters

Croisier, Esteban és Galand (1976): Lehetséges egy kettős szűrővel szétosztott és lemintavételezett jelet tökéletesen visszaállítani még akkor is, ha a szűrők nem ideálisak.

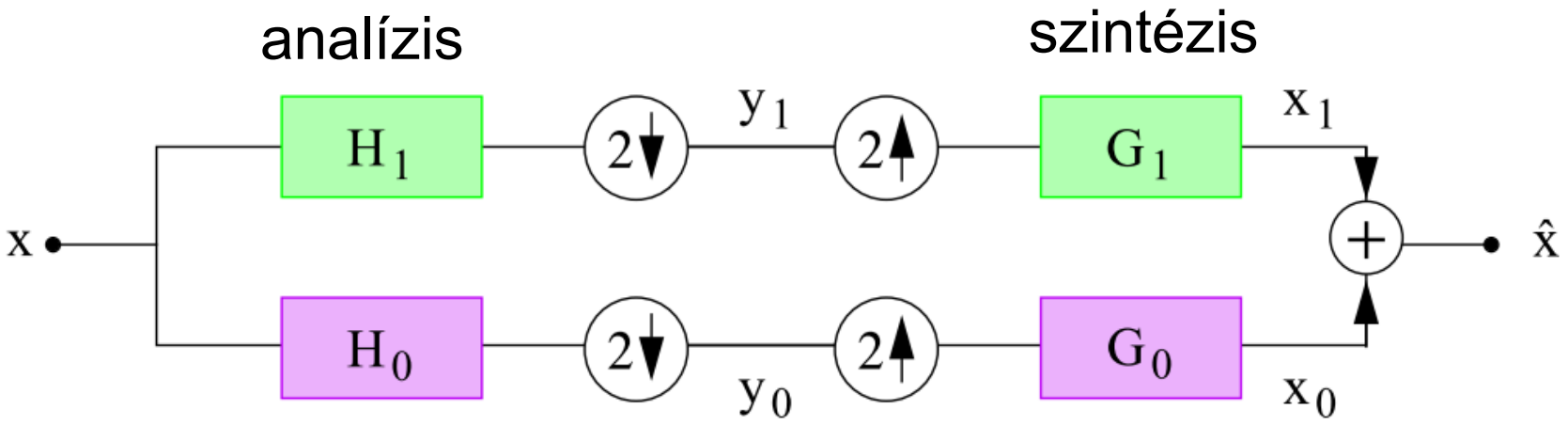
$$G_0H_0 + G_1H_1 = I$$



Kettős tükörszűrők (QMF)

Croisier, Esteban és Galand (1976): Lehetséges egy kettős szűrővel szétosztott és lemintavételezett jelet tökéletesen visszaállítani még akkor is, ha a szűrők nem ideálisak.

$$G_0H_0 + G_1H_1 = I$$



Kettős tükörszűrő példa

konvolúciós szűrők

$$\mathbf{h} = (\ldots, h_{-1}, h_0, h_1, h_2, \ldots) \qquad \mathbf{x}(\ldots, x_{-1}, x_0, x_1, x_2, \ldots)$$

$$\mathbf{y} = \mathbf{h} * \mathbf{x} \qquad y_n = \sum_k h_k x_{n-k}$$

a legegyszerűbb szűrő: $\mathbf{h} = (h_0, h_1) = (\frac{1}{2}, \frac{1}{2})$

$$y_n = \frac{1}{2} x_n + \frac{1}{2} x_{n-1}$$

a konvolúció mátrixa:

A konvolúció mátrixa

$$H = \begin{bmatrix}
\ddots & & \ddots & \ddots & \ddots & & & & & & & & & \\
 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & \\
\cdots & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & \cdots \\
 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & \\
 & & & & & & & & & \ddots & \ddots & & & \ddots
\end{bmatrix}$$

A szűrt jelből az eredetit nem lehet visszaállítani

Mi szükséges ahhoz, hogy a szűrés megfordítható legyen?

A szűrés megfordítása

Az egymás melletti számok **átlaga** mellett a **különbségükre** is szükség van a megfordításhoz

A különbség képzés az alábbi konvolúciós szűrést jelenti:

$$\mathbf{g} = (\tfrac{1}{2}, -\tfrac{1}{2})$$

Most már valóban megfordítható a szűrés. Ha

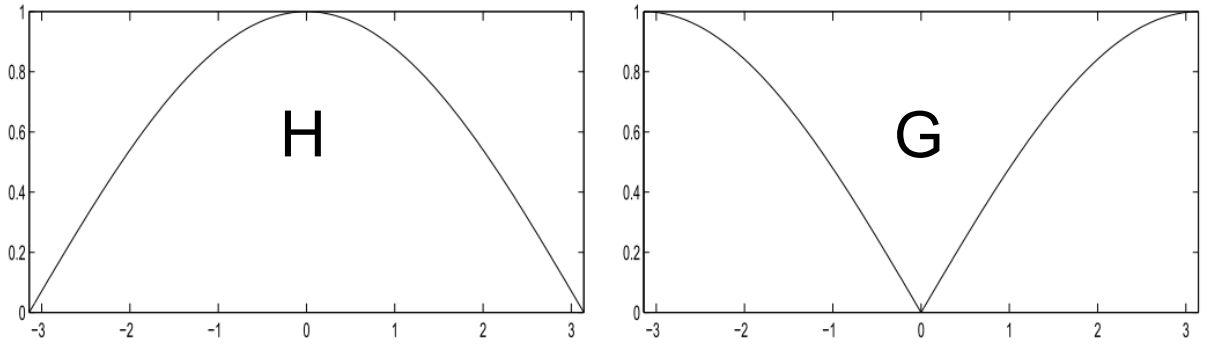
$$y_n = \frac{1}{2}x_n + \frac{1}{2}x_{n-1} \qquad z_n = \frac{1}{2}x_n - \frac{1}{2}x_{n-1}$$

akkor $x_n = y_n + z_n$ $x_{n-1} = y_n - z_n$

A szűrőpár viselkedése

$$G = \begin{bmatrix} \ddots & & \ddots & \ddots & \ddots & & & & & & & & & \\ & 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ & 0 & 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & \\ \dots & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & \dots \\ & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & \\ & & & & & & & & & \ddots & \ddots & & & \ddots \end{bmatrix}$$

A **H** és **G** szűrők alul- és felülátereszt˝ok:



A szűrés és megfordítása

$$\begin{bmatrix} H \\ _ \\ G \end{bmatrix} \cdot \mathbf{x} = \begin{bmatrix} \mathbf{y} \\ _ \\ \mathbf{z} \end{bmatrix}$$

A **H** és **G** szűrőkkel végzett szűrés megfordítható.

$$\begin{aligned}
&\vdots && \vdots \\
y_1 - z_1 &= \frac{x_1 + x_0}{2} - \frac{x_1 - x_0}{2} &&= x_0 \\
y_1 + z_1 &= \frac{x_1 + x_0}{2} + \frac{x_1 - x_0}{2} &&= x_1 \\
y_2 - z_2 &= \frac{x_2 + x_1}{2} - \frac{x_2 - x_1}{2} &&= x_1 \\
y_2 + z_2 &= \frac{x_2 + x_1}{2} + \frac{x_2 - x_1}{2} &&= x_2 \\
y_3 - z_3 &= \frac{x_3 + x_2}{2} - \frac{x_3 - x_2}{2} &&= x_2 \\
y_3 + z_3 &= \frac{x_3 + x_2}{2} + \frac{x_3 - x_2}{2} &&= x_3 \\
&\vdots && \vdots
\end{aligned}$$

A szűrés megfordítása

Az inverzió redundáns – nem kell az **összes** y_n, z_n érték x_n visszaállításához!

y_2, z_2 pl. elhagyható:

Sőt, **H** és **G** minden második sora fölösleges – alulmintavételezünk

$$
\begin{aligned}
&\vdots && \vdots \\
y_1 - z_1 &= \frac{x_1 + x_0}{2} - \frac{x_1 - x_0}{2} &&= x_0 \\
y_1 + z_1 &= \frac{x_1 + x_0}{2} + \frac{x_1 - x_0}{2} &&= x_1 \\
y_2 - z_2 &= \frac{x_2 + x_1}{2} - \frac{x_2 - x_1}{2} &&= x_1 \\
y_2 + z_2 &= \frac{x_2 + x_1}{2} + \frac{x_2 - x_1}{2} &&= x_2 \\
y_3 - z_3 &= \frac{x_3 + x_2}{2} - \frac{x_3 - x_2}{2} &&= x_2 \\
y_3 + z_3 &= \frac{x_3 + x_2}{2} + \frac{x_3 - x_2}{2} &&= x_3 \\
&\vdots && \vdots
\end{aligned}
$$

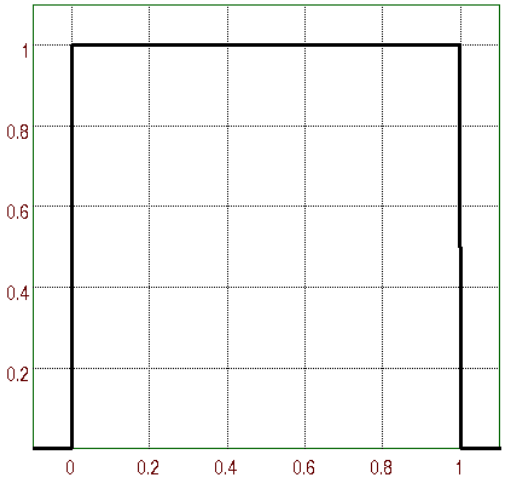
A transzformáló mátrix

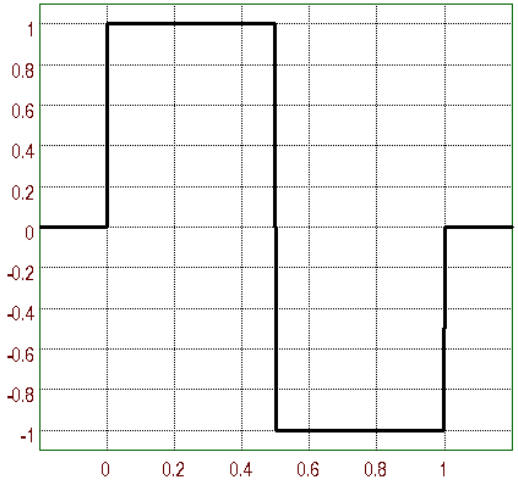
Ha $\mathbf{x} = (x_0, \ldots, x_N)$,

akkor az eredmény az alábbi mátrix:

$$\tilde{W}_N = \left[\begin{array}{cccccccccc}
\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & & & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & & \ldots & 0 & 0 \\
\vdots & & & & \ddots & & & & & \vdots \\
0 & 0 & 0 & 0 & \ldots & & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\
\hline
-\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & & & 0 & 0 \\
0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & & \ldots & 0 & 0 \\
\vdots & & & & \ddots & & & & & \vdots \\
0 & 0 & 0 & 0 & \ldots & & 0 & 0 & -\frac{1}{2} & \frac{1}{2}
\end{array}\right]$$

Haar wavelet





Haar skálázó függvény

Haar wavelet

A transzformáló mátrix inverze

A mátrix inverze egyszerű, mert

$$x_n = y_n + z_n$$

$$x_{n-1} = y_n - z_n$$

$$\tilde{W}_N^{-1} = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ \vdots & & \ddots \; \vdots & \vdots & & \ddots \; \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

Ortogonalizáció

A transzformáló mátrix már majdnem ortogonális:

$$\tilde{W}_N^T = \frac{1}{2}\tilde{W}_N^{-1}$$

Ha $\tilde{W}_N$ -et $\sqrt{2}$ -vel szorozzuk, ortogonális lesz.

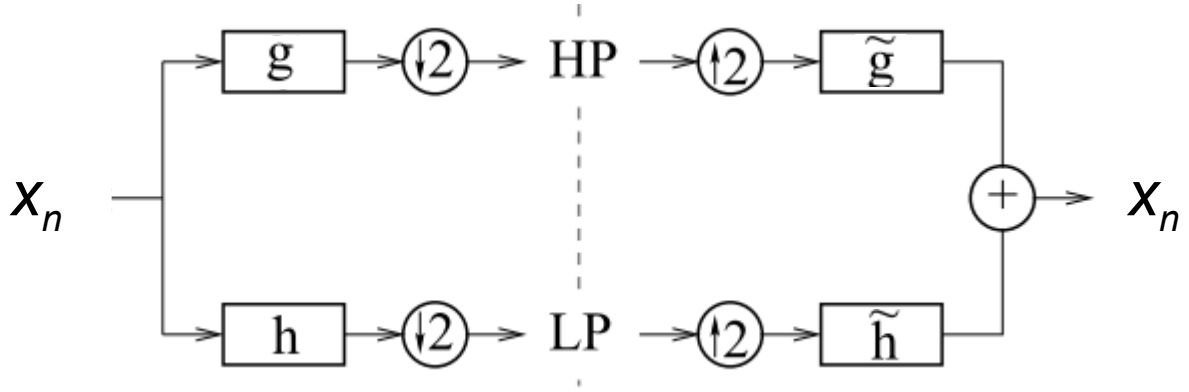
$$W_N = \begin{bmatrix} H \\ \hline G \end{bmatrix} = \left[\begin{array}{cccccc|c} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 & & 0 & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & & 0 & 0 \\ \vdots & & & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & \dots & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \hline -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 & & 0 & 0 \\ 0 & 0 & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & & 0 & 0 \\ \vdots & & & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & \dots & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{array}\right]$$

Diszkrét Haar wavelet transzformáció

A W_N elnevezése: Diszkrét Haar wavelet transzformáció

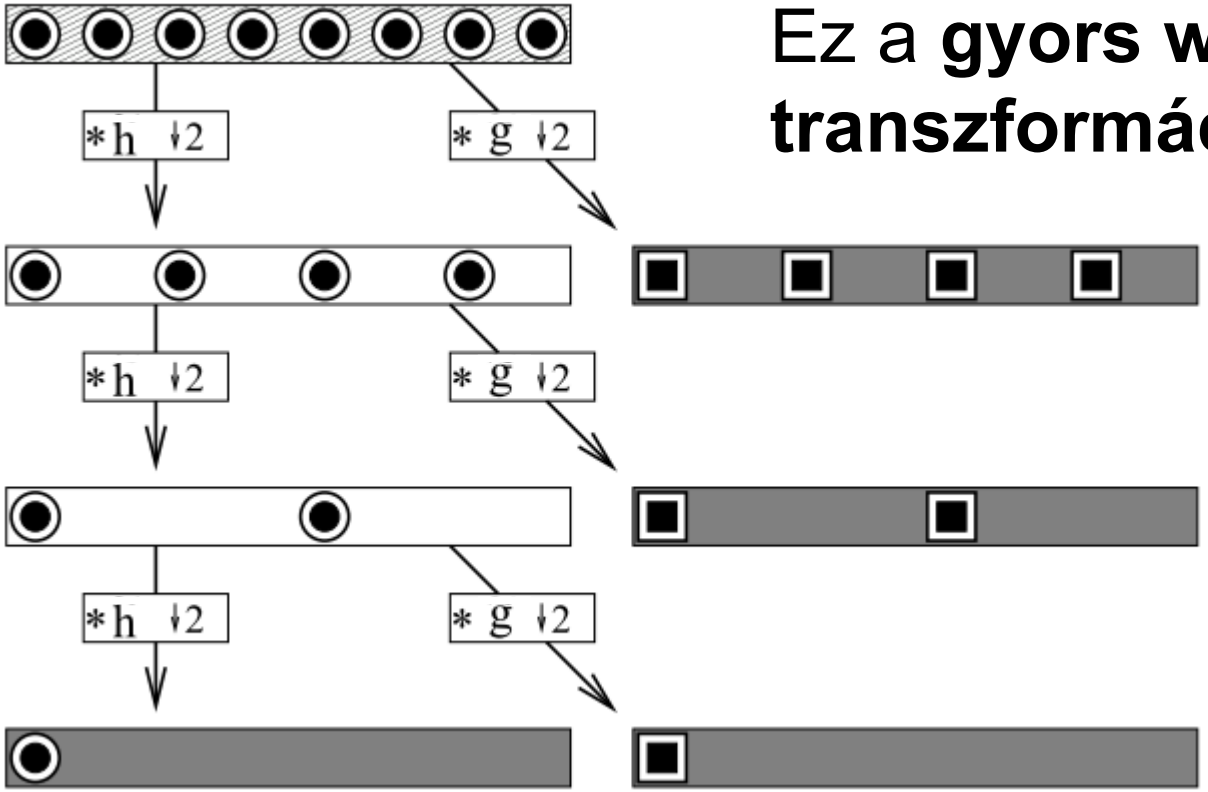
A Haar szűrő: $\mathbf{h} = \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \right)$ $H(\omega) = \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} e^{i\omega}$

A ***h*** és ***g*** kettős tükör szűrő, mert x_n torzítás nélkül visszaállítható, a szűrés tökéletesen megfordítható.



Szűrőcsoport, FWT

A szűrés és lemintavételezés ciklikusan ismételhető.



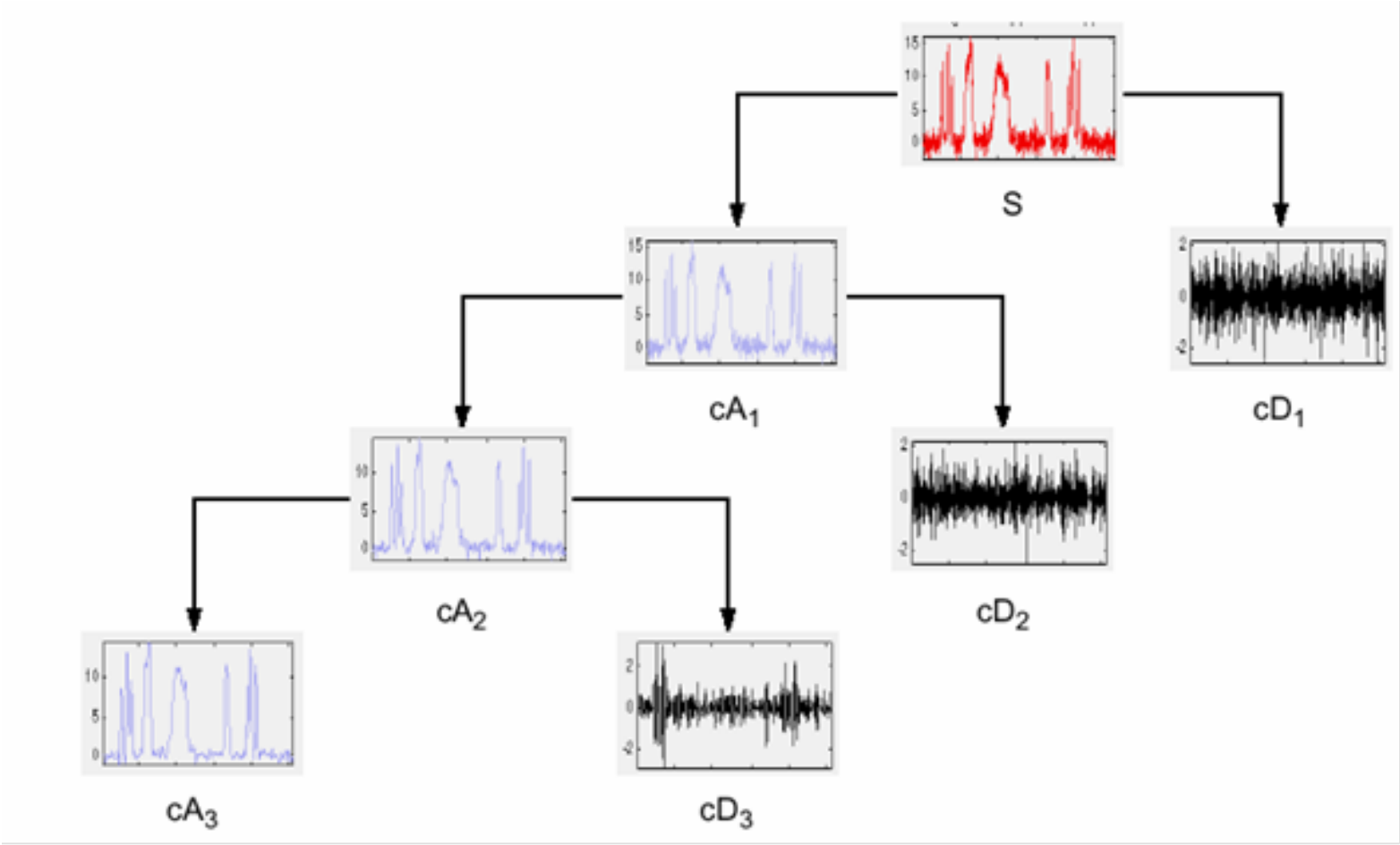
Ez a **gyors wavelet transzformáció** (FWT)

Sokskálás felbontás, MRA

Egy $L^2(\boldsymbol{R})$-ben adott jelet egymásba ágyazott V_j alterekben fokozatosan közelítünk. Ezeket az altereket egyetlen $\varphi(t)$ **skálázó függvény** eltolt és átskálázott változatai generálják.

Ha V_0 a szakaszonként konstans függvények lineáris tere, $\boldsymbol{Z}$-ben ugrásokkal, a $\varphi(t) = \chi_{[0,1)}(t)$ karakterisztikus függvény és eltolt változatai V_0 ortonormális bázisa.

Sokskálás felbontás, MRA



2D Wavelet transzformáció

Van egy *N* x *N*-es méretű *A* mátrixunk (*N* páros)
Mi lesz *A* wavelet transzformáltja?

Direkt szorzat transzformáció

Soroként transzformáljuk az *A* mátrixot, majd az így transzformált mátrixot oszloponként is transzformáljuk

Közelítés, tömörítés

Egy tipikus jel esetében a szomszédos minták erősen korreláltak

Ez esetben a felülátereszt ő ***G*** szűrő eredményeként előálló wavelet együtthatók kicsik lesznek és nagy részük sok esetben el is hagyható.

Kézenfekvő alkalmazás teljesen kitöltött mátrixú egyenletrendszerek megoldása wavelet prekondicionálással, vagy integrál operátorok közelítése.

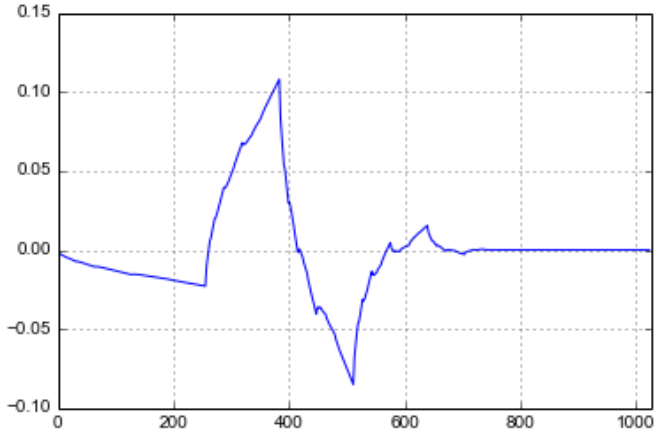
Interaktív Jupyter munkafüzet (Python kóddal)

https://nbviewer.jupyter.org/github/gyulat/Wavelet/blob/master/DWT.ipynb

```
## inverz 2D Daubechies2 wavelet transzformáció
def invdaub2(X):
    a = np.copy(X)
    a = invdaub1(a).T
    return invdaub1(a).T
```

Most is nézzük meg, hogyan néz ki a Daubechies2 elemző wavelet.

In [11]:
```
v=np.zeros((2,1024))
v[:,4]=1
iv=invdaub1(v)
plt.plot(iv[1,:])
plt.grid()
plt.xlim([0,1025])
plt.show()
```



Elvégezzük a kép Daubechies2 DWT elemzését a már megírt függvényekkel, kiszámítjuk a diszkrét wavelet együtthatók mátrixát.

Haar wavelet transzformáció

```
## Haar wavelet transzformáció
## v mátrix is lehet, ekkor mindegyik sorát transzformáljuk
def haar1(v):
    n=v.shape[1]
    i=n
    while i>1:
        av=(v[:,:i:2]+v[:,1:i:2])/2.0
        dv=v[:,:i:2]-av
        v[:,:i]=np.hstack((av,dv))
        i=i/2
    return v

## sorok majd oszlopok transzformálása: 2D Haar WT
def haar2(X):
    a = np.copy(X)
    a = haar1(a).T
    return haar1(a).T
```

Inverz Haar WT

```
## inverz Haar wavelet transzformáció
def invhaar1(v):
    n=v.shape[1]
    i=2
    while i<=n:
        av=np.copy(v[:,0:i/2])
        dv=np.copy(v[:,i/2:i])
        v[:,:i:2]=av+dv
        v[:,1:i:2]=av-dv
        i=2*i
    return v

def invhaar2(X):
    a = np.copy(X)
    a = invhaar1(a).T
    return invhaar1(a).T
```

Daubechies 1D, 2D wavelet transzformáció

```
## DAUB2 wavelet transzformáció
def daub1(v):
    n=v.shape[1]
    i=n
    c=daubechies2()
    while i>2:
    #    ov=1:2:(i-1); ev=2:2:i
    #    ov1=3:2:(i-1)|1; ev2=4:2:i|2
        ov1=range(2,i,2)
        ov1.append(0)
        ev2=range(3,i,2)
        ev2.append(1)
        av=(c[0]*v[:,:i:2]+c[1]*v[:,1:i:2]+c[2]*v[:,ov1]+c[3]*v[:,ev2])
        dv=(c[3]*v[:,:i:2]-c[2]*v[:,1:i:2]+c[1]*v[:,ov1]-c[0]*v[:,ev2])
        v[:,:i]=np.hstack((av,dv))
        i=i/2
    return v


## sorok majd oszlopok transzformálása: 2D DAUB2 WT
def daub2(X):
    a = np.copy(X)
    a = daub1(a).T
    return daub1(a).T
```

Daubechies inverz 1D, 2D wavelet transzformáció

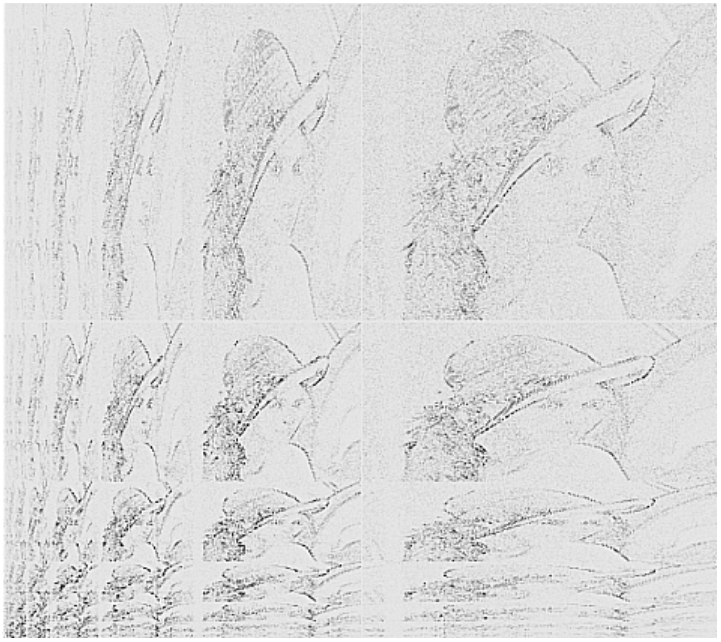
```
## inverz 1D DAUB2 wavelet transzformáció
def invdaub1(v):
    n=v.shape[1]
    i=4
    c=daubechies2()
    while i<=n:
        ivx=range(0,i/2-1)
        iv1=[i/2-1]
        iv1=iv1+ivx
        av=np.copy(v[:,0:i/2])
        dv=np.copy(v[:,i/2:i])
        v[:,:i:2]=c[0]*av+c[2]*av[:,iv1]+c[3]*dv+c[1]*dv[:,iv1]
        v[:,1:i:2]=c[1]*av+c[3]*av[:,iv1]-c[2]*dv-c[0]*dv[:,iv1]
        i=2*i
    return v

## inverz 2D DAUB2 wavelet transzformáció
def invdaub2(X):
    a = np.copy(X)
    a = invdaub1(a).T
    return invdaub1(a).T
```

Léna tesztkép traszformációja

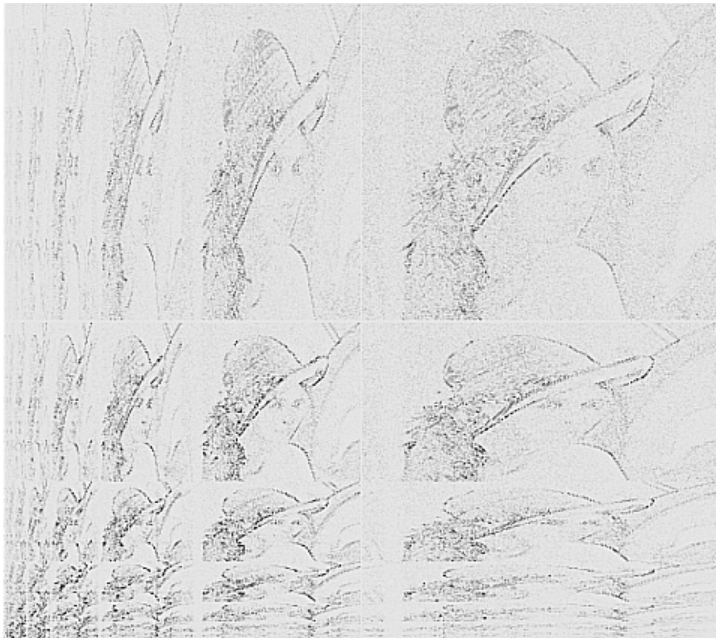
Haar wavelet DWT





Léna tesztkép tömörítése

Haar wavelet iDWT, 14% együtthatóval

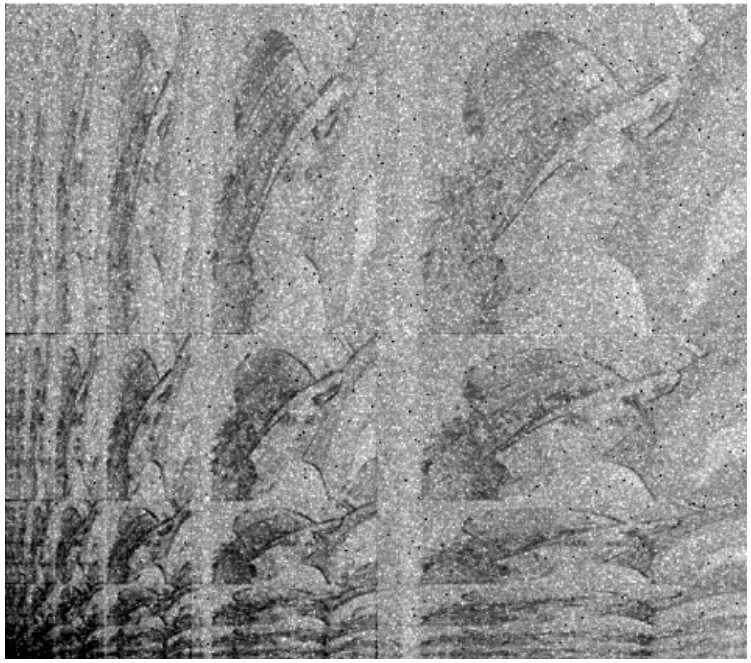




Léna tesztkép traszformációja

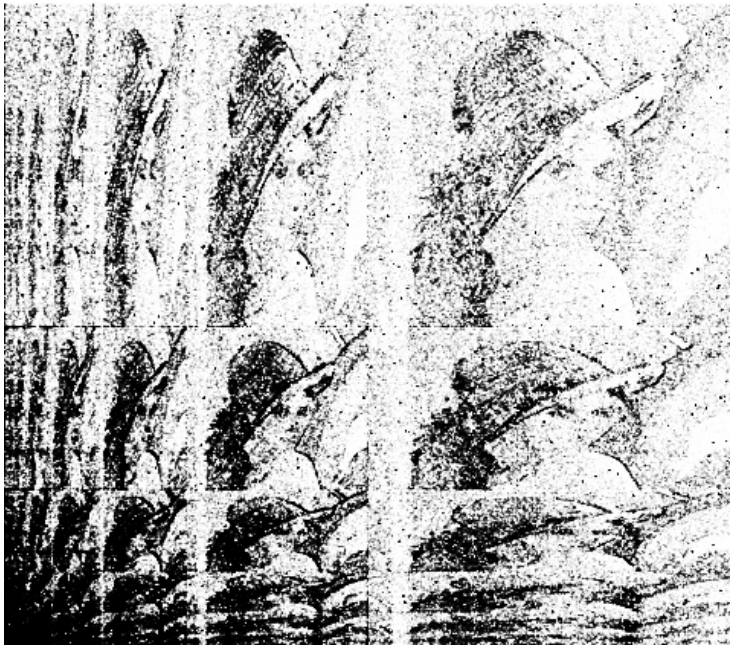
Daubechies DAUB2 wavelet DWT





Léna tesztkép tömörítése

DAUB2 wavelet iDWT, 7% együtthatóval





Lineáris egyenletrendszerek

Teljesen kitöltött mátrixú rendszer

$$Ax = b$$

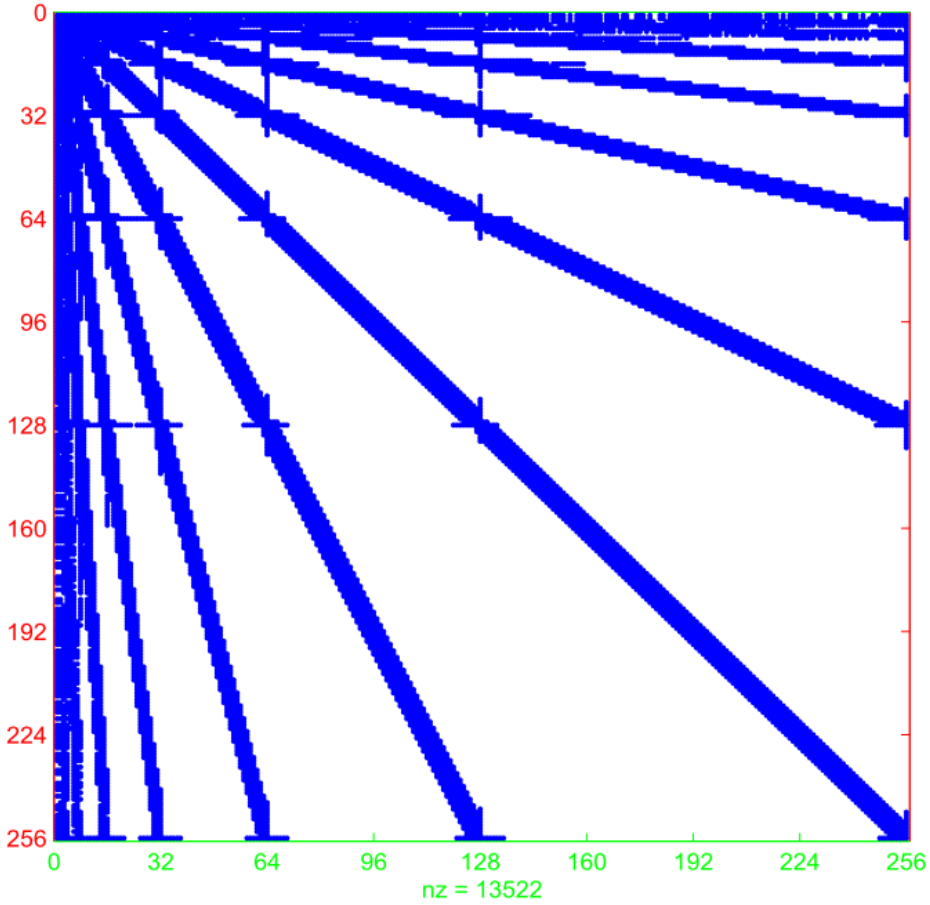
A wavelet transzformáció mátrixa:

$$\overline{W}_\nu = \begin{pmatrix}
c_0 & c_1 & c_2 & c_3 & & & & & & \\
d_0 & d_1 & d_2 & d_3 & & & & & & \\
 & & c_0 & c_1 & c_2 & c_3 & & & & \\
 & & d_0 & d_1 & d_2 & d_3 & & & & \\
 & & & & \ddots & \ddots & \ddots & \ddots & & \\
 & & & & \ddots & \ddots & \ddots & \ddots & & \\
 & & & & & & c_0 & c_1 & c_2 & c_3 \\
 & & & & & & d_0 & d_1 & d_2 & d_3 \\
c_2 & c_3 & & & & & & & c_0 & c_1 \\
d_2 & d_3 & & & & & & & d_0 & d_1
\end{pmatrix}$$

Transzformált együttható mátrix

A transzformáció: $\widetilde{A} = WAW^{\top}$

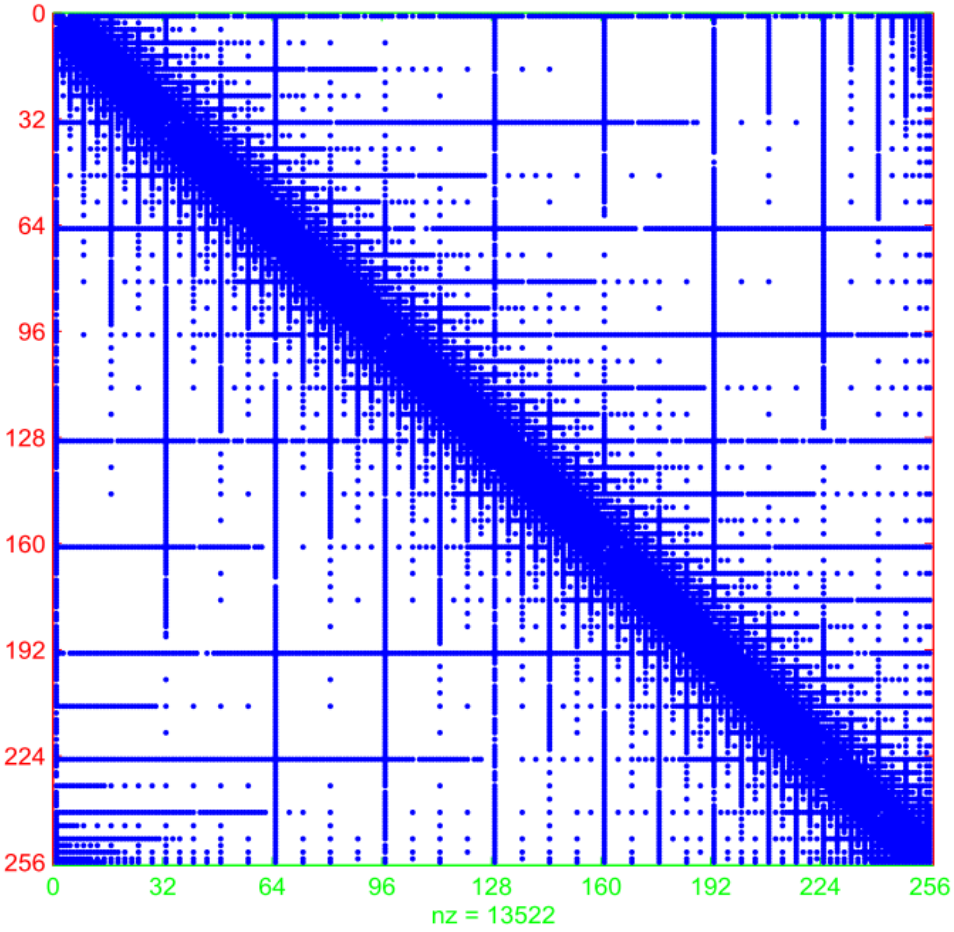
A transzformált mátrix:
(8 transzform. szint,
10^{-4} küszöbérték)



A permutált mátrix

Átrendezés után: $P(WAW^\top)P^\top$

átlósan domináns alakra történő átrendezés



Operátor tömörítés

Integrál egyenlet:

$$g(x) = \int_R K(x,y) f(y) dy \quad f \in H \quad g \in H$$

Megoldás Galerkin-módszerrel: *f* közelítő f_n megoldását nem a *H*, hanem a közelítő H_n alterében keressük meg:

$$f_n(x) = \sum_{i=1}^{n} \alpha_i \varphi_i(x)$$

Galerkin-módszer

A közelítő megoldás maradéka,

$$r(x) = \int_{\mathbb{R}} K(x,y) \sum_{i=1}^{n} \alpha_i \varphi(y) dy$$

ortogonális H_n-re:

$$\int_{\mathbb{R}} r(x)\varphi_k(x)dx = 0 \quad k = 1, \ldots, n$$

A Galerkin-egyenleteket kell megoldani α_i-re.

$$\sum_{i=0}^{N} a_{ki}\alpha_i = b_k \qquad k = 0, \ldots N$$

$$a_{ki} := \int_R \int_R K(x,y)\varphi_i(y)dy\varphi_k(x)dx \qquad b_k := \int_R g(x)\varphi_k(x)dx$$

Konjugált gradiens módszer

A CG módszer konvergencia sebessége a μ max. és λ min. sajátértékektől függ:

$$\rho = \frac{\mu - \lambda}{\mu + \lambda}$$

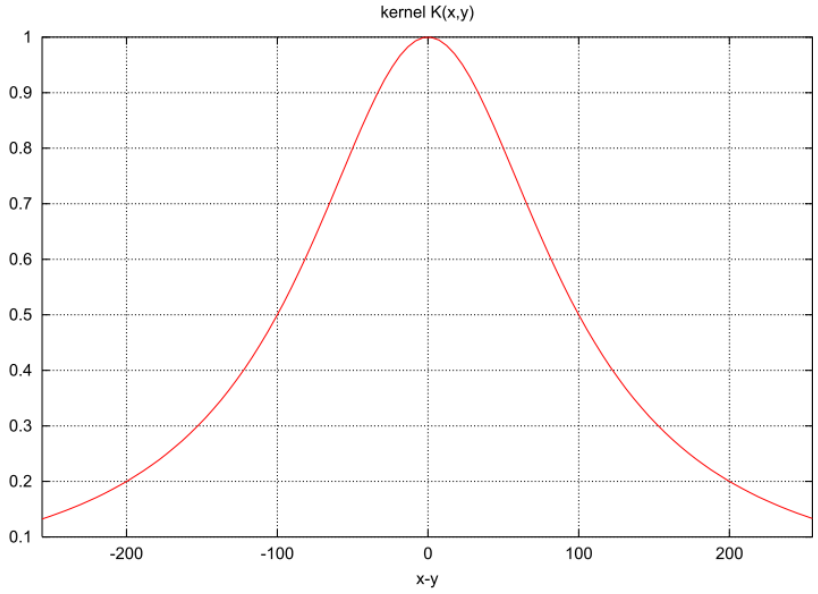
Ha ez közel van 1-hez, a konvergencia igen lassú

Wavelet bázissal a megoldás javítható.

Példa

Az alábbi integrál egyenletet oldjuk meg:

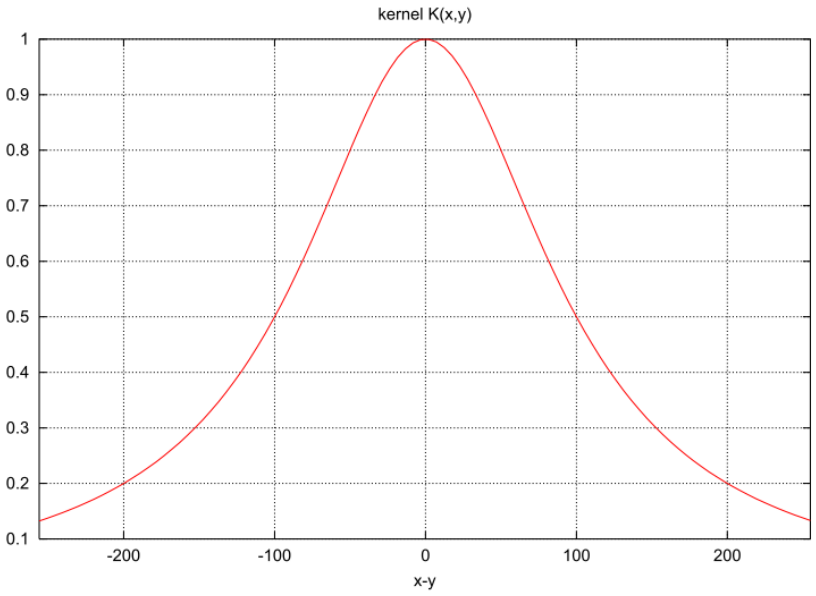
$$\int_{\mathbb{R}} K(x,y)f(y)dy = g(x) \qquad K(x,y) = \frac{1}{1+\alpha(x-y)^2} \quad , \quad \alpha = 10^{-4}$$



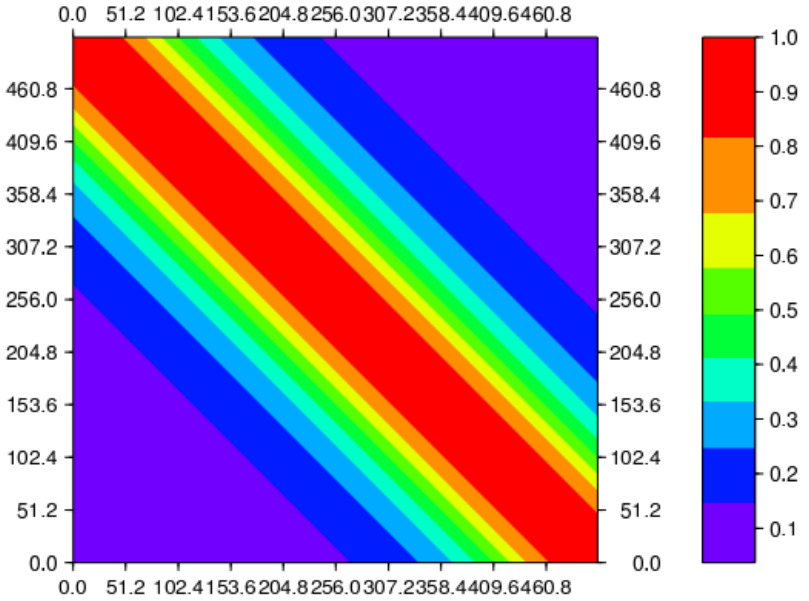
Példa

Az alábbi integrál egyenletet oldjuk meg:

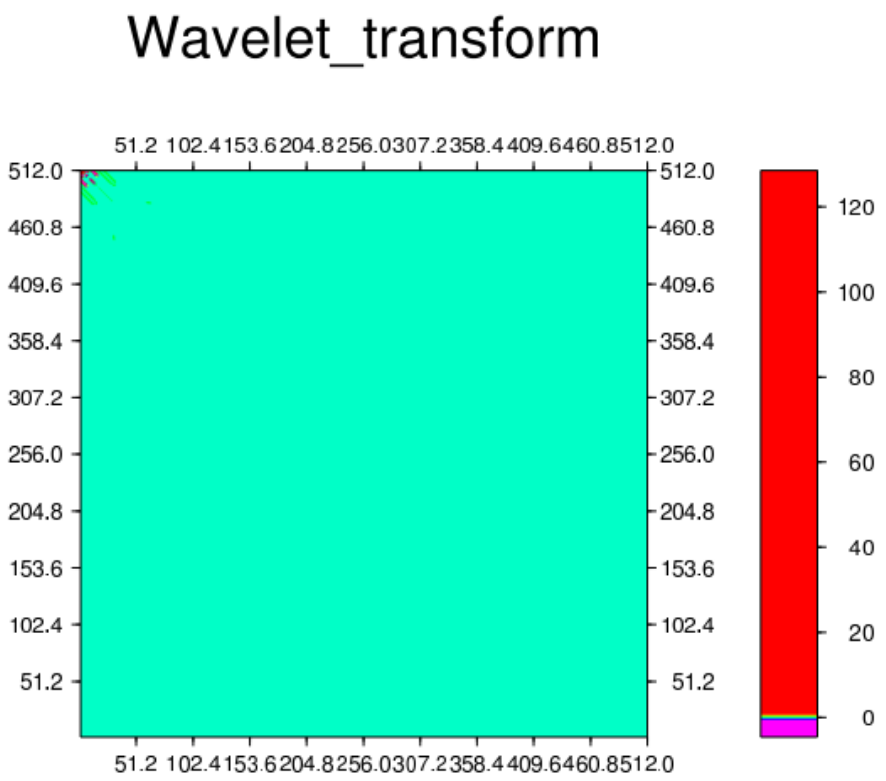
$$\int_{\mathbb{R}} K(x,y)f(y)dy = g(x) \qquad K(x,y) = \frac{1}{1+\alpha(x-y)^2} \quad , \quad \alpha = 10^{-4}$$

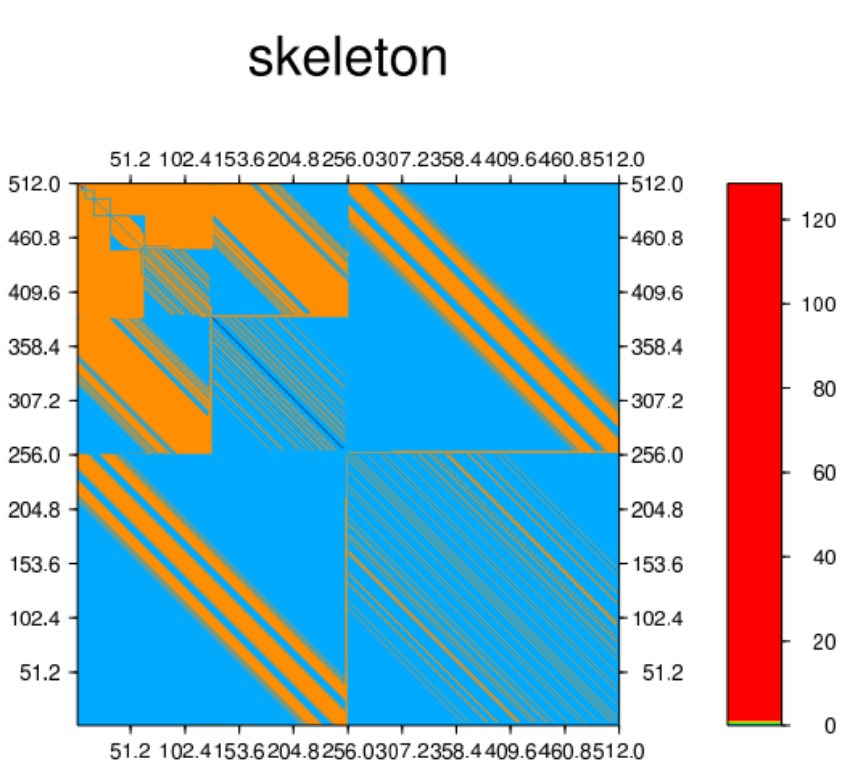


Galerkin-matrix



Wavelet prekondicionálás





Konvergencia

