内容安排的基本逻辑

绪论	研究方法初步	随机信号基础	时域	频域	时频域	综合运用
1 课程目标、内容安排、生物医学信号、基本工具	2 科学问题、实验设计、数据采集、建模与仿真	3 概率统计（自学环节） 4 随机过程	5a 互相关及其应用 5b 叠加平均及其应用	6 确定信号的FFT、随机信号功率谱估计	7 STFT、Gabor变换、CWT 8 DWT 9 小波变换应用 10 综合练习（自学环节）	11 综合案例 – 从科学问题到基于逻辑推理和量化证据的结论 12 考试

第八讲 离散小波变换的基本原理

讨论要点

- 连续小波变换存在的困难

- 离散小波变换的基本原理
  - Mallat 快速算法
  - 案例：Haar 小波变换
  - 基于频域滤波视角的观察

- 离散小波变换的主要MATLAB函数
  - dwt, wavedec, swt

连续小波变换

- 设信号$f(t) \in L^2(R)$，母小波$\psi(t)$经过伸缩和平移得到的小波函数簇$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$，信号$f(t)$的小波变换为

$$W_f(a,b) = \frac{1}{\sqrt{a}}\int_{-\infty}^{\infty}\psi\left(\frac{t-b}{a}\right)f(t)dt$$

  - a是尺度因子，为正实数，确定窗口长度
  - b是平移因子，可是任意实数，确定窗口的中心位置
  - t，a和b的取值均为连续变量，上述变换成为连续小波变换（continuous wavelet transform, CWT）

- 尺度系数a与窗口长度的关系
  - 记得高斯窗的数学形式即容易理解
  - 尺度系数a变小，则时间上压缩，时间窗口变小，获得高的时间分辨率，有助于分析快速的波形变化（高频成分）
  - 尺度系数a变大，则时间上拉伸，时间窗口变长，获得高的频率分辨率，有助于分析较小的频率变化（低频成分）

连续小波变换的步骤

（1）选择一个小波函数，尺度系数为a_1，将这个小波与待分析信号起点对齐

（2）计算小波函数与信号的卷积，获得小波变换系数C，小波系数越大，该段信号与小波函数波形越像

（3）将小波函数沿时间轴右移一个时间单位b，重复第2步，获得新的变换系数

（4）重复第2-3步，直到信号结束，获得尺度a_1下的所有小波系数

（5）改变小波尺度系数成a_2，重复第1-4步，获得尺度系数a_2下的所有小波系数

（6）重复步骤1-5，获得所有感兴趣的尺度系数a_i（i=1, 2, … …, N）对应的小波系数

连续小波变换的一些问题

- 理论上，在连续小波变换中，尺度因子$a_i>0$，位移因子b_i，和时间t_i都是连续的
- 对于离散数字信号而言，b_i和t_i的最小单位是采样间隔，由采样率决定，a_i仍可以是任意正实数
- 存在的问题
  - a_i的取值如何确定？ a_i和a_{i+1}的间隔理论上可以任意大小
  - b_i的步长在理论上可以是≥1个采样间隔
  - 过度密集采样a_i和b_i会导致巨大的计算量
  - 过度密集采样会产生大量冗余数据
  - 不足的采样导致信息的损失

离散小波变换

- 对小波变换 $W_f(a,b)=\frac{1}{\sqrt{a}}\int_{-\infty}^{\infty}\psi\left(\frac{t-b}{a}\right)f(t)dt$ 中的尺度*a*与时移b参数离散化，获得离散小波变换（discrete wavelet transform, dwt)
- 离散化方案
  - 尺度按照幂级数离散化 $a = a_0^j$，j是整数，可正，可负
  - 对时移b的离散化为关于尺度系数的函数，例如$b = ka_0^j$，如$k=b/a=b/a_0^j$

$$W_f(j,k)=\frac{1}{\sqrt{a_0^j}}\int_{-\infty}^{\infty}\psi\left(a_0^{-j}t-k\right)f(t)dt$$

dwt的实质——对时-频空间的采样

$$W_f(j,k)=\frac{1}{\sqrt{a_0^j}}\int_{-\infty}^{\infty}\psi\left(a_0^{-j}t-k\right)f(t)dt$$

$a=a_0^j$，j确定小波**频率**窗口的位置；$b=ka_0^j$，k确定小波**时间**窗口的位置

* 尺度和位移均随着信号成分的频率特性自动调节，例如，对于低频成分，尺度系数增大，因为低频信号变化相对较慢，可以降低时间采样时间，减少信息冗余，节省计算时间

* 理想情况下，取a_i和b_i使每一个小波系数分别表征时频图所有格子内的信息，既无冗余，且完整表征原信号的内容

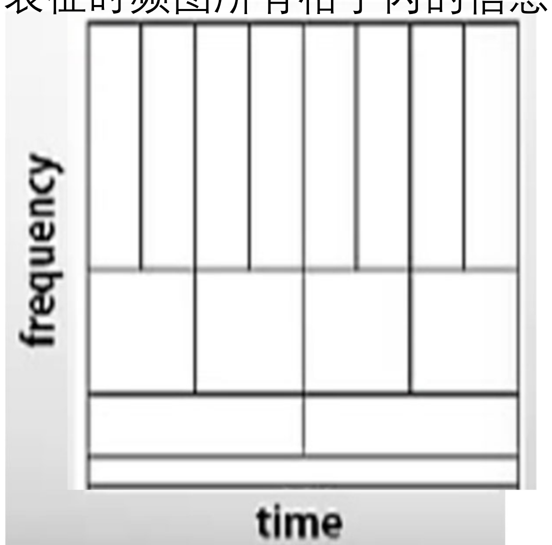
a_0>1的情形，

$a_1=a_0^1$

$a_2=a_0^2$

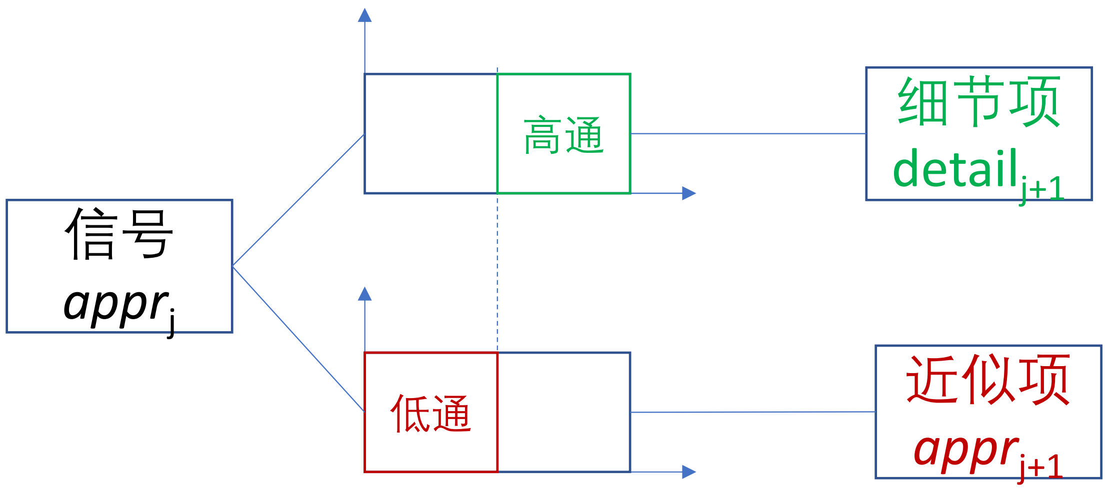
$a_3=a_0^3$

Increasing a_i



信号近似项和信号细节项的分解

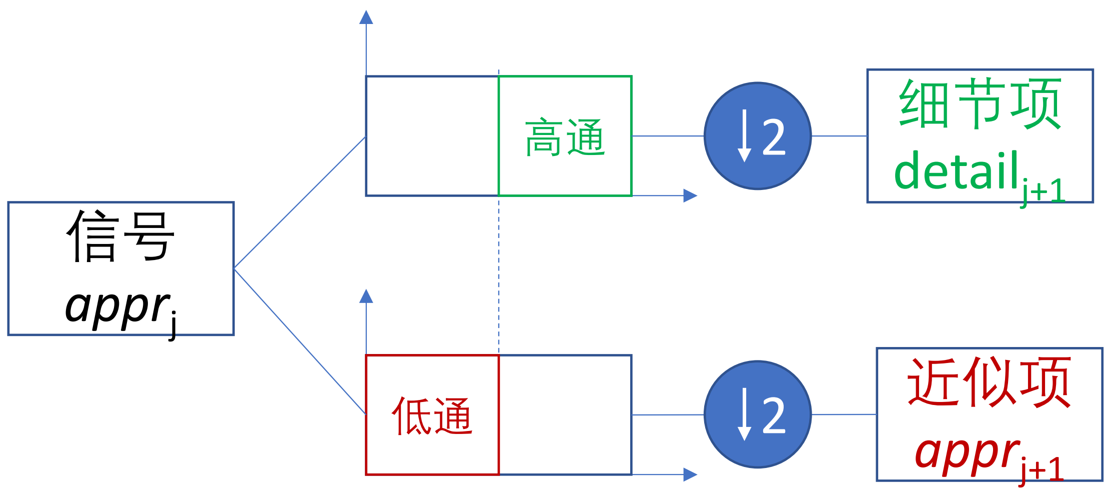
- Gabor函数本质上是一个带通滤波器，可提取信号中一系列频带内的成分（等价于分解）
- 更一般地，小波变换可实现信号从高频到低频的逐级分解
  - 高频部分，信号细节项(detail)
  - 低频部分，信号近似项(approximation)



dwt的Mallet快速算法

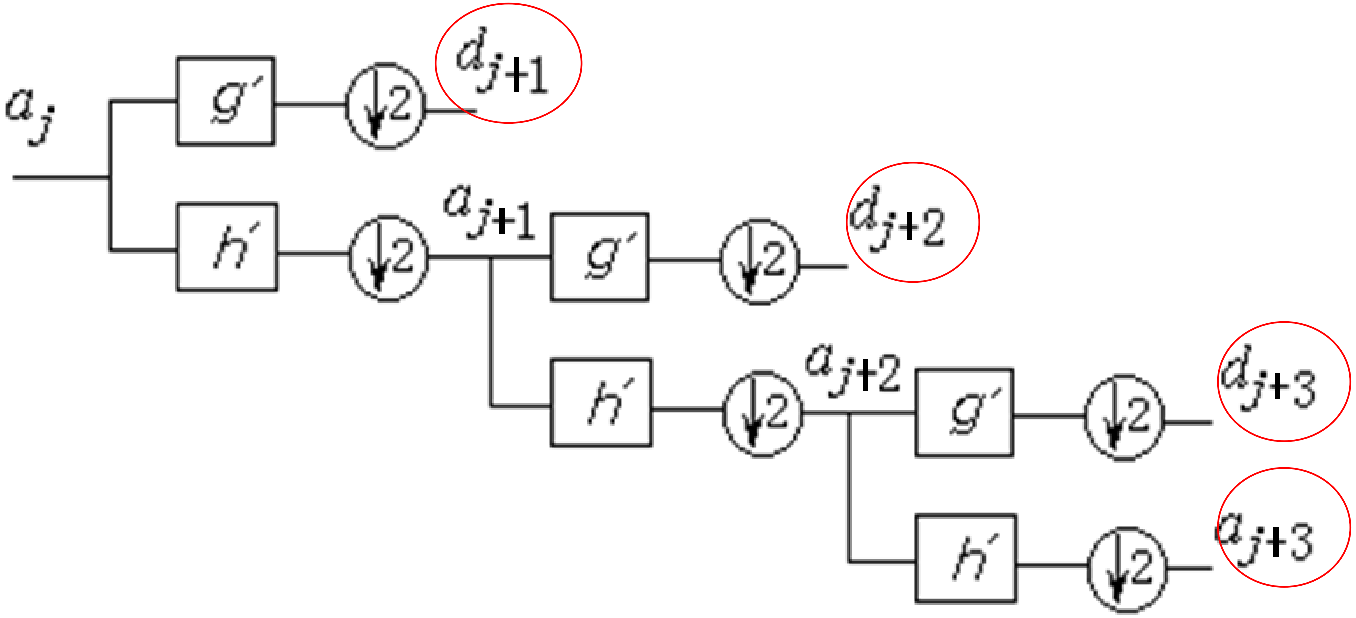
- 步骤

  （1）信号通过低通和高通滤波，获得近似项和细节项
  （2）对滤波结果进行基-2的下采样
  （3）对上述近似项再次进行低通和高通滤波
  （4）迭代步骤（2）和（3）直到获得预定的分解次数



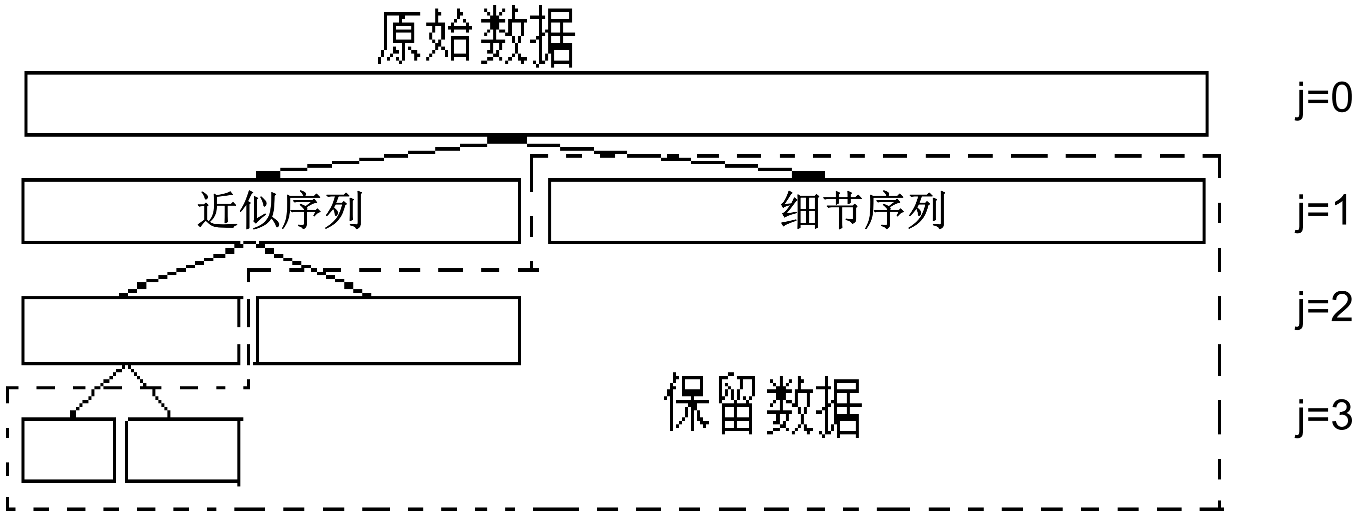
* 对信号的下采样等价为对小波的尺度拉伸

多级分解



表示2倍下采样，在两个样本数据中取一个

Mallet算法的数据量不变性质

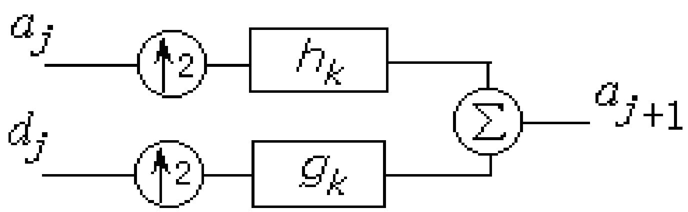


塔式数据
塔式算法

离散小波逆变换实现信号重构

- Inverse discrete wavelet transform (idwt)

- 是信号分解的逆过程
  - 上采样，综合

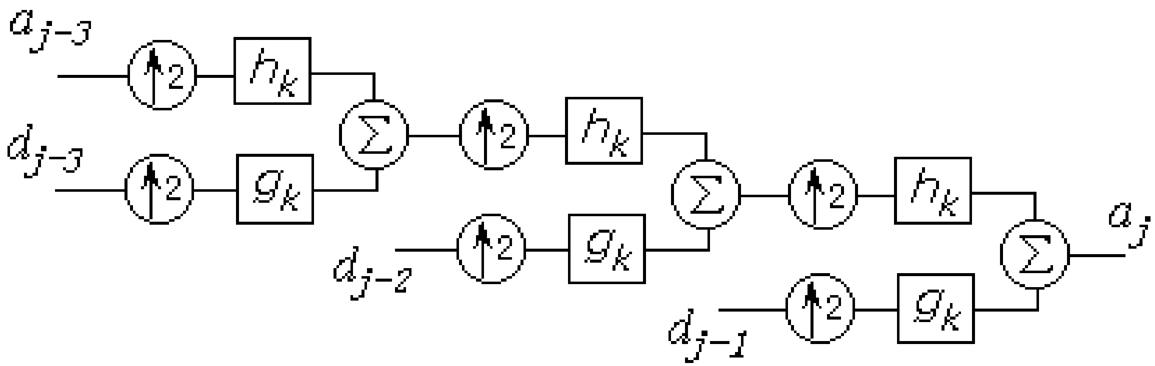


重构算法

$\uparrow 2$ 表示2倍上采样，在两个数据中插入一个“0”

多级重构 (idwt)

- 信号分解的逆过程
  - 上采样，综合
  - 迭代



多级重构算法

表示2倍上采样，在两个样本数据中插入一个“0”

案例：Haar小波变换

- 滤波器组系数
  - 分解低通 $[1 \quad 1]/\sqrt{2}$
  - 分解高通 $[-1 \quad 1]/\sqrt{2}$
  - 重构低通 $[1 \quad 1]/\sqrt{2}$
  - 重构高通 $[1 \quad -1]/\sqrt{2}$

分解过程

循环卷积　　　　　　　　偶数项2-下采样

- 低通，获得近似项

$$[8 \quad 10 \quad 12 \quad 15 \quad 12 \quad 14 \quad 12 \quad 5]/\sqrt{2} \rightarrow [10 \quad 15 \quad 14 \quad 5]/\sqrt{2}$$

- 高通，获得细节项

$$[-4 \quad 2 \quad -4 \quad 1 \quad 2 \quad -4 \quad 6 \quad 1]/\sqrt{2} \rightarrow [2 \quad 1 \quad -4 \quad 1]/\sqrt{2}$$

一维原始信号

$$[6 \quad 4 \quad 8 \quad 7 \quad 5 \quad 9 \quad 3 \quad 2]$$

分解低通　$[1 \quad 1]/\sqrt{2}$

分解高通　$[-1 \quad 1]/\sqrt{2}$

分解的具体过程

- 考虑边界效应，首先将原始信号左边补充最后边的数字2，得到新的向量 [**2** 6 4 8 7 5 9 3 2]
- 在原序列对应位置上的卷积结果
  - 对于分解低通滤波，6位置对应的结果为 $(2+6)/\sqrt{2}$，以此类推，得到[8 10 12 15 12 14 12 5]/$\sqrt{2}$，然后做偶数位置上的基-2下采样，得到[10 15 14 5]/$\sqrt{2}$
  - 对于分解高通滤波，6位置对应的结果为 [2*1 + 6*(-1)]/$\sqrt{2}$，以此类推，得到[-4 2 -4 1 2 -4 6 1]/$\sqrt{2}$；然后做基-2下采样，得到[2 1 -4 1]/$\sqrt{2}$

重构过程

- 二插值　　　　右循环卷积

$$\begin{bmatrix} 0 & 10 & 0 & 15 & 0 & 14 & 0 & 5 \end{bmatrix}/\sqrt{2} \rightarrow \begin{bmatrix} 10 & 10 & 15 & 15 & 14 & 14 & 5 & 5 \end{bmatrix}/2$$

$$\begin{bmatrix} 0 & 2 & 0 & 1 & 0 & -4 & 0 & 1 \end{bmatrix}/\sqrt{2} \rightarrow \begin{bmatrix} 2 & -2 & 1 & -1 & -4 & 4 & 1 & -1 \end{bmatrix}/2$$

- 求和，重构原始信号

$$\begin{bmatrix} 6 & 4 & 8 & 7 & 5 & 9 & 3 & 2 \end{bmatrix}$$

重构低通　$[1 \quad 1]/\sqrt{2}$

重构高通　$[1 \quad -1]/\sqrt{2}$

一维原始信号

$$\begin{bmatrix} 6 & 4 & 8 & 7 & 5 & 9 & 3 & 2 \end{bmatrix}$$

重构的具体过程

- 对近似项[10 15 14 5]/$\sqrt{2}$的处理：
  - 做基-2上采样: [0 10 0 15 0 14 0 5]/$\sqrt{2}$
  - 右循环补充一位：[0 10 0 15 0 14 0 5 **0**]/$\sqrt{2}$,
  - 重构低通滤波：[10 10 15 15 14 14 5 5]/2
- 对细节项[2 1 -4 1]/$\sqrt{2}$的处理：
  - 做基-2上采样: [0 2 0 1 0 -4 0 1]/$\sqrt{2}$
  - 右循环补充一位：[0 2 0 1 0 -4 0 1 **0**]/$\sqrt{2}$
  - 重构高通滤波：第一个位置上的结果为 (-1*0+1*2)/2，以此类推，得到[2 -2 1 -1 -4 4 1 -1]/2
- 上述两项结果之和：
  - [12 8 16 14 10 18 6 4]/2，即原来的序列[6 4 8 7 5 9 3 2]

在频率域观察小波分解和重构

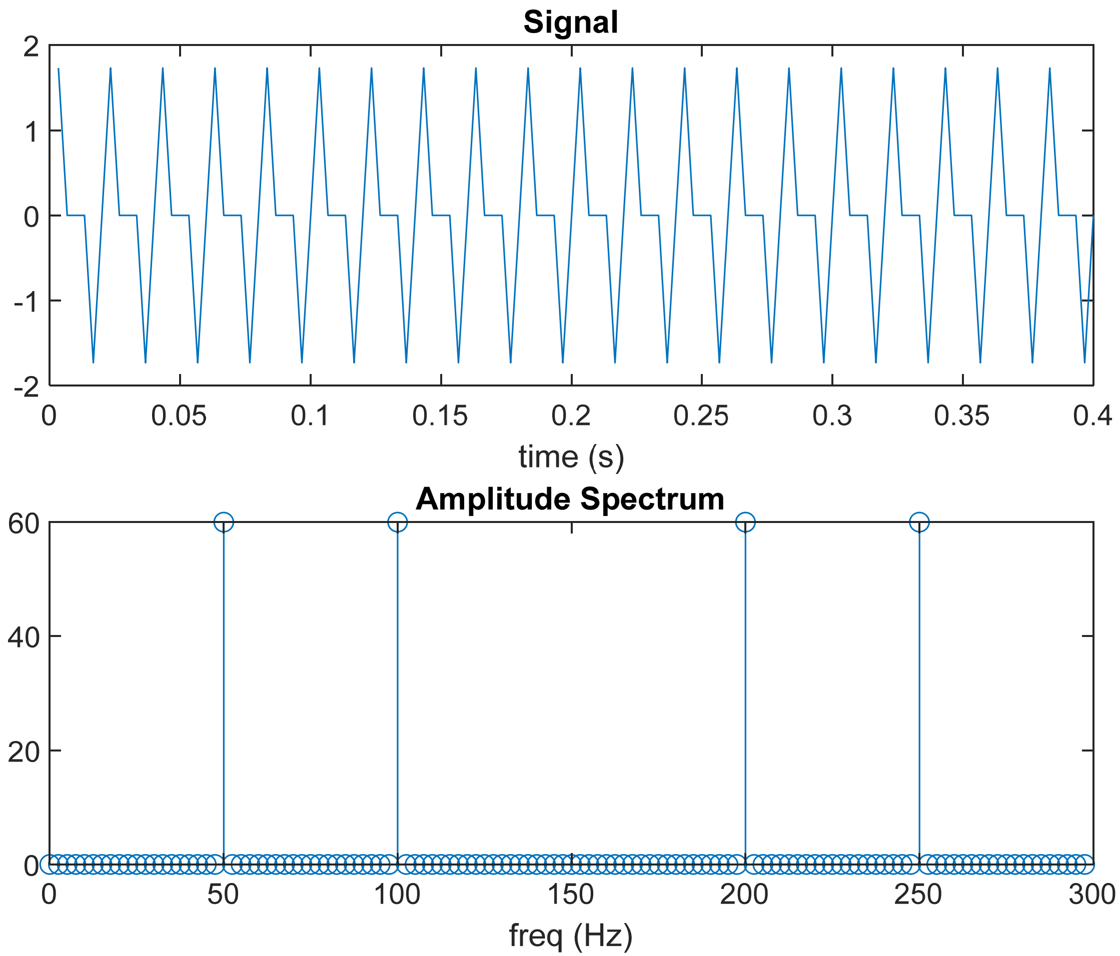
- 仿真时域内待分解的信号，长度为N
- 计算小波对应的高通和低通滤波器的频域特性
- 采用频域滤波的方法，实现信号的分解

仿真待分解信号

- % 1. 正弦波参数定义
- f1 = 50; % 频率1
- f2 = 100; % 频率2
- fs = 2*(f1+f2); % 采样频率
- Ts = 1/fs; % 采样间隔
- N = 120; % 采样点数

- % 2. 仿真两个正弦波的叠加
- n = 1:N;
- y = sin(2*pi*f1*n*Ts)+sin(2*pi*f2*n*Ts); % 正弦波混合
- tTick = n/fs;
- fTick = (n-1)/N*fs;

- % 3. 显示两个正弦波的时域和频域特性
- figure(1); subplot(2,1,1); plot(tTick, y);
- title('Signal'); xlabel(‘time (s)’);
- subplot(2,1,2); stem(fTick, abs(fft(y)));
- title('Amplitude Spectrum'); xlabel(‘freq (Hz)’);


Signal
2
1
0
-1
-2
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
time (s)
Amplitude Spectrum
60
40
20
0
0
50
100
150
200
250
300
freq (Hz)

小波滤波器特性

- %% 2.小波滤波器谱分析
- h = wfilters('db30','l'); % 低通
- g = wfilters('db30','h'); % 高通

- h = [h, zeros(1, N-length(h))]; % 补零与原始数据等长，便于频域内滤波
- g = [g, zeros(1, N-length(g))]; % 补零与原始数据等长，便于频域内滤波

- figure(2);
- subplot(2,1,1);
- stem(fTick, abs(fft(h))); %stem函数用于绘制火柴梗图
- title('Low-pass Filter(V_{0})');

- subplot(2,1,2);
- stem(fTick, abs(fft(g)));
- title('High-pass Filter(W_{0})');

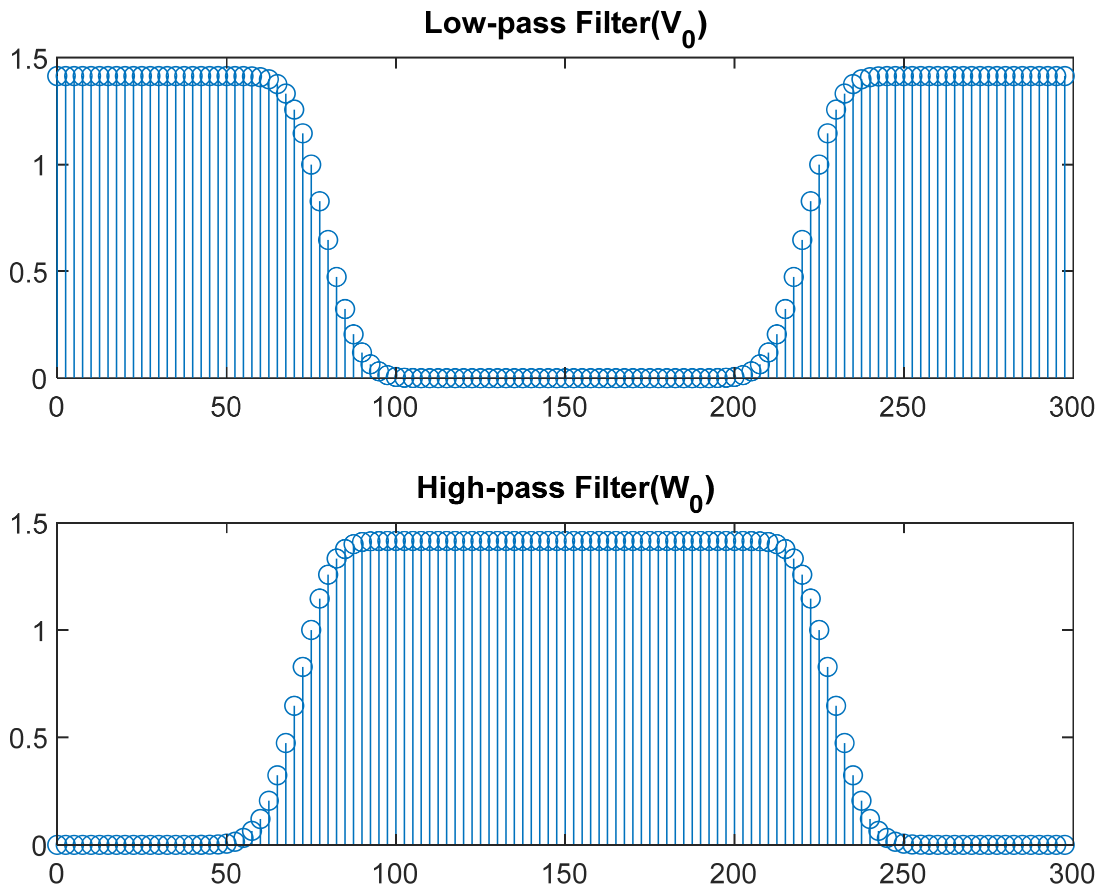

Low-pass Filter(V_0)
1.5
1
0.5
0
0
50
100
150
200
250
300
High-pass Filter(W_0)
1.5
1
0.5
0
0
50
100
150
200
250
300

figure2

频率域内实现小波变换

- %% 3.MALLAT分解算法
  % 时域内的圆周卷积，对应的快速傅里叶变换

- sig1=ifft(fft(y).*fft(h)); % 低通(低频分量)
- sig2=ifft(fft(y).*fft(g)); % 高通(高频分量)

- figure(3); % 信号图
- subplot(2,1,1)
- plot(tTick, real(sig1));
- title('Low-frequency Component')

- subplot(2,1,2)
- plot(tTick, real(sig2));
- title('High-frequency Component')

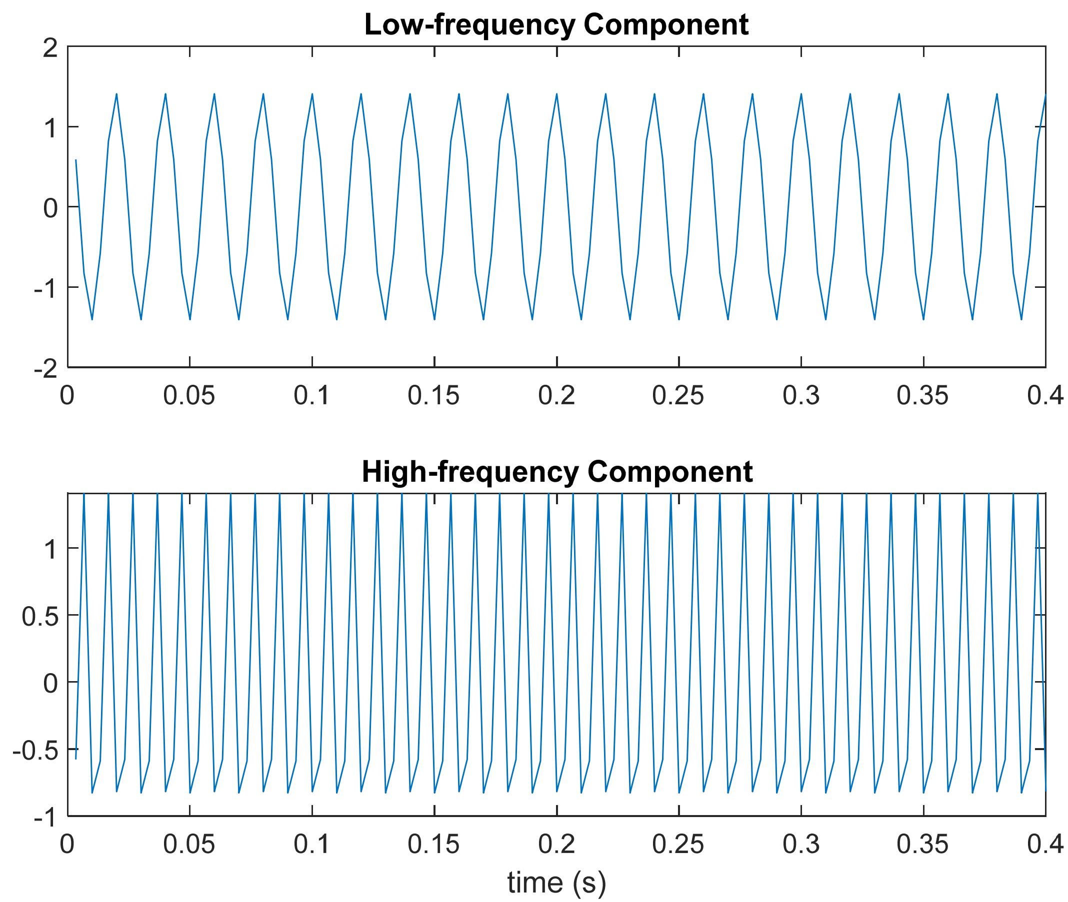

Low-frequency Component
2
1
0
-1
-2
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
High-frequency Component
1
0.5
0
-0.5
-1
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
time (s)

figure3

分解成分对应的傅里叶谱

- %%频谱图
- figure(4); % 频谱图
- subplot(2,1,1)
- stem(fTick, abs(fft(sig1)));
- title('Amplitude Spectrum of Low-frequency Component')

- subplot(2,1,2)
- stem(fTick, abs(fft(sig2)));
- title('Amplitude Spectrum of High-frequency Component')

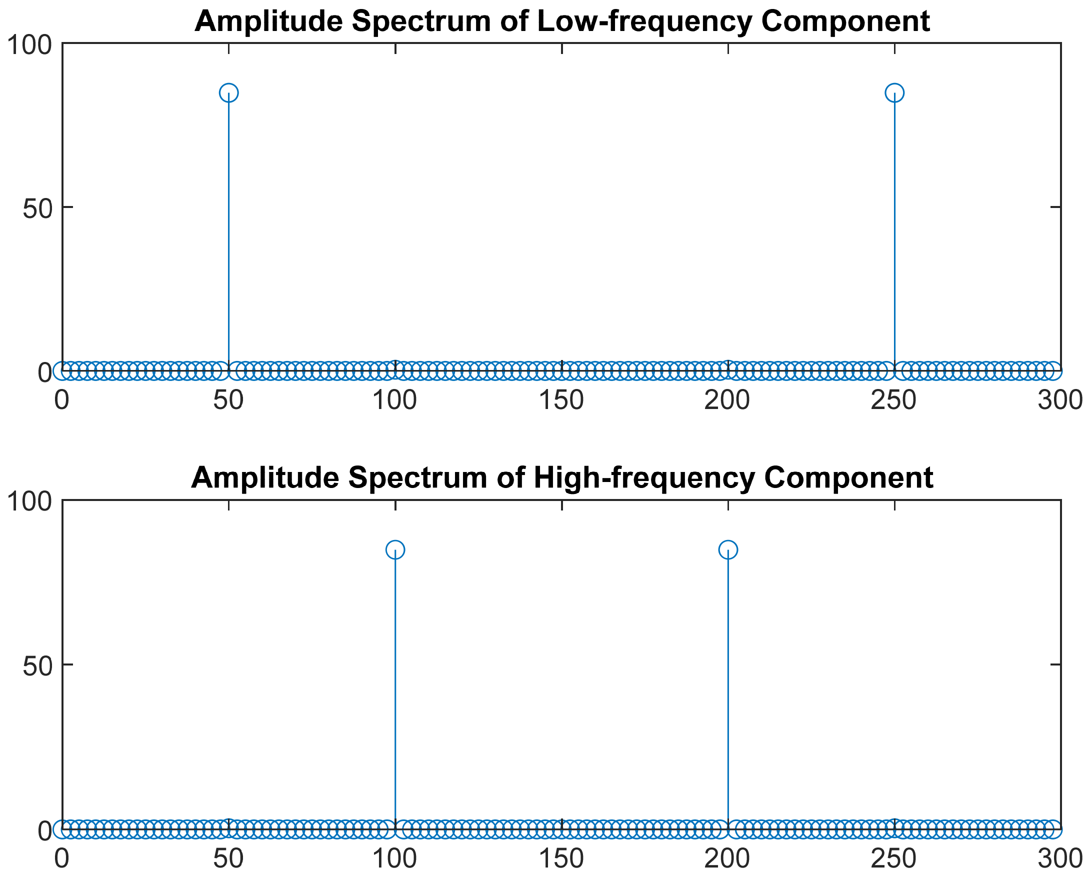

Amplitude Spectrum of Low-frequency Component
100
50
0
0
50
100
150
200
250
300
Amplitude Spectrum of High-frequency Component
100
50
0
0
50
100
150
200
250
300

figure4

重构算法的频域实现

- %% 4.MALLAT重构算法
- sig1=dyaddown(sig1); % 2抽取
- sig2=dyaddown(sig2); % 2抽取

- sig1=dyadup(sig1);   % 2插值，补零
- sig2=dyadup(sig2);   % 2插值
- sig1=sig1(1,[1:N]); % 去掉最后一个零
- sig2=sig2(1,[1:N]); % 去掉最后一个零

- hr=h(end:-1:1);   % 重构低通
- gr=g(end:-1:1);   % 重构高通
- hr=circshift(hr',1)';   % 位置调整圆周右移一位
- gr=circshift(gr',1)';   % 位置调整圆周右移一位

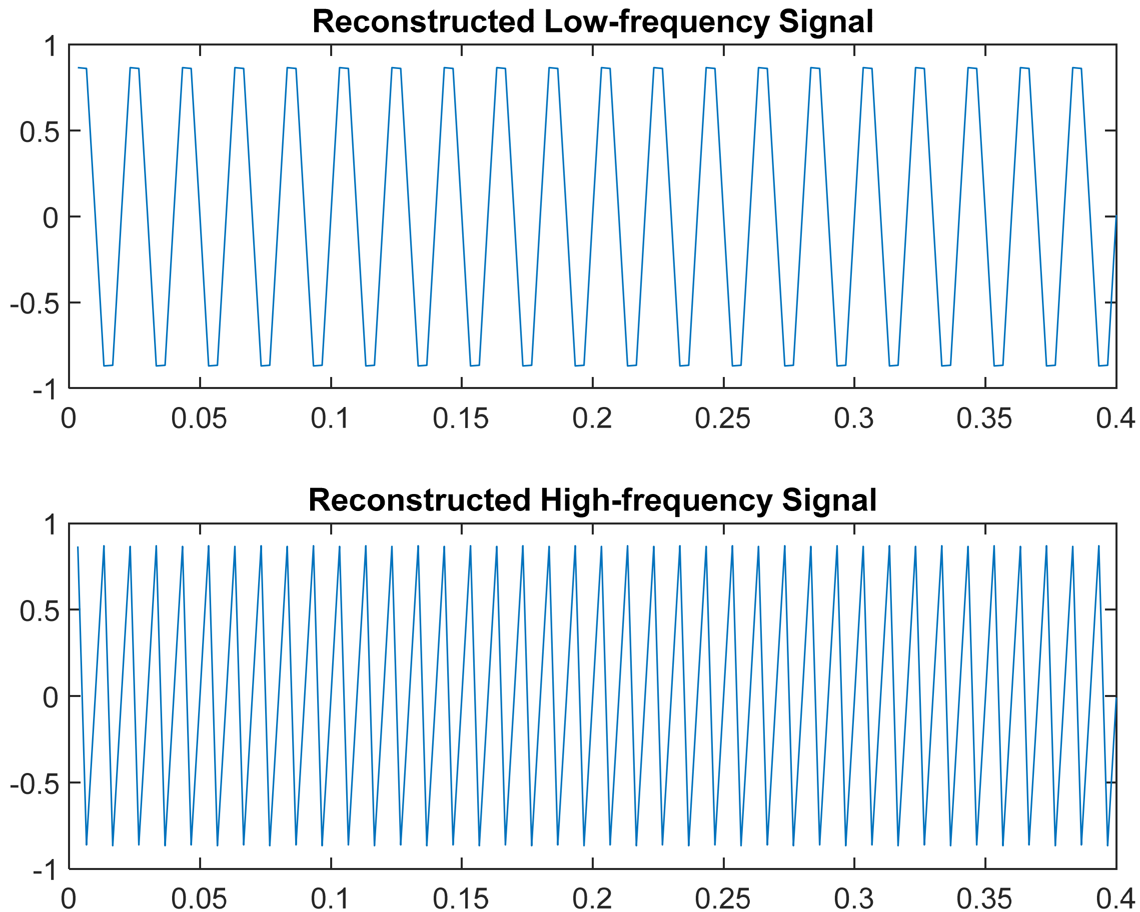
- sig1=ifft(fft(hr).*fft(sig1)); % 低频分量，卷积定理
- sig2=ifft(fft(gr).*fft(sig2)); % 高频分量
- sig=sig1+sig2; % 重构的信号（高频+低频分量）

比较重构信号和原始信号

```
%% 5.比较
figure(5);
subplot(2,1,1);  plot(tTick, real(sig1));
title('Reconstructed Low-frequency Signal');
subplot(2,1,2);  plot(tTick, real(sig2));
title('Reconstructed High-frequency Signal');

figure(6);
subplot(2,1,1);  stem(fTick, abs(fft(sig1)));
title('Spectra of the Reconstructed Low-frequency Signal');
subplot(2,1,2);  stem(fTick, abs(fft(sig2)));
title('Spectra of the Reconstructed High-frequency Signal');

figure(7)
plot(tTick, real(sig),'r','linewidth',2);
hold on;  plot(tTick, y, ‘b-’ );
legend('Reconstructed Signal', 'Original Signal')
title('Comparisons between Original Signal and Reconstructed Signal')
```



Reconstructed Low-frequency Signal
1
0.5
0
-0.5
-1
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
Reconstructed High-frequency Signal
1
0.5
0
-0.5
-1
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4

figure5

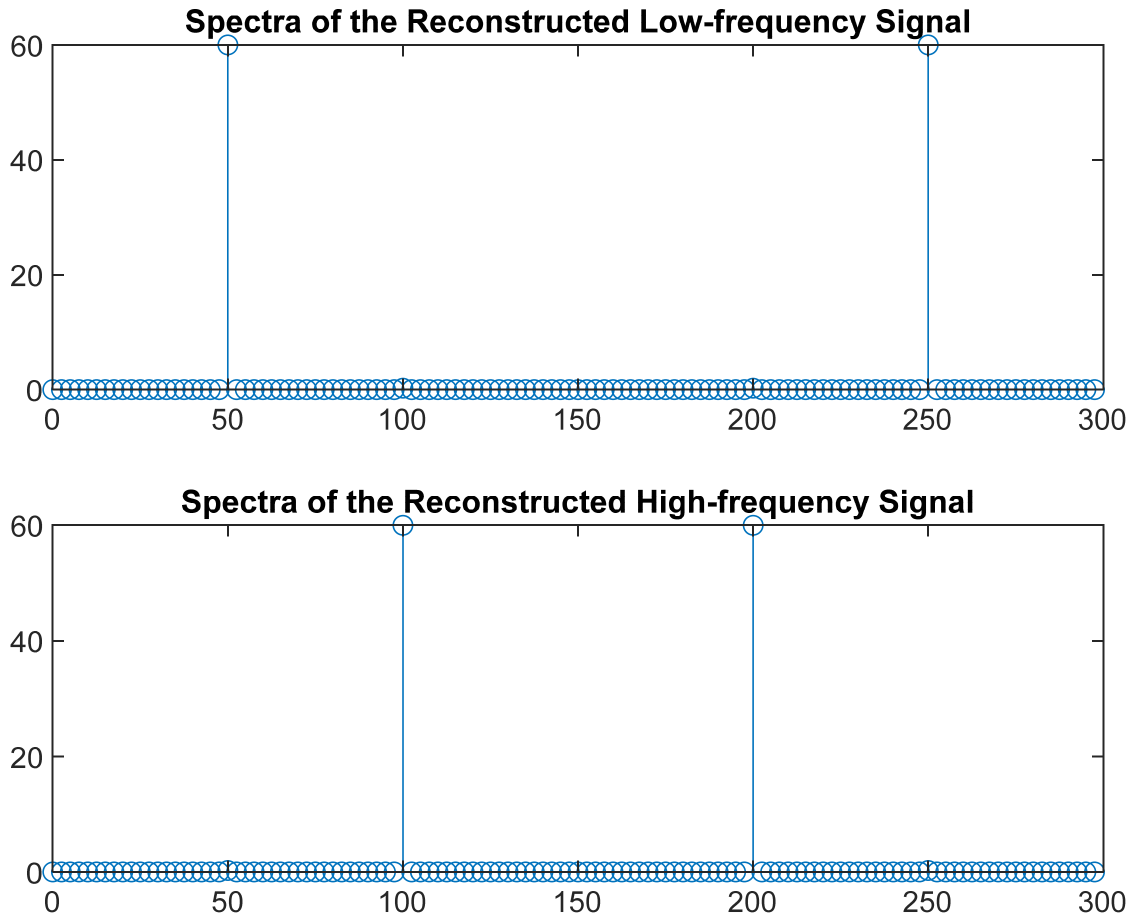
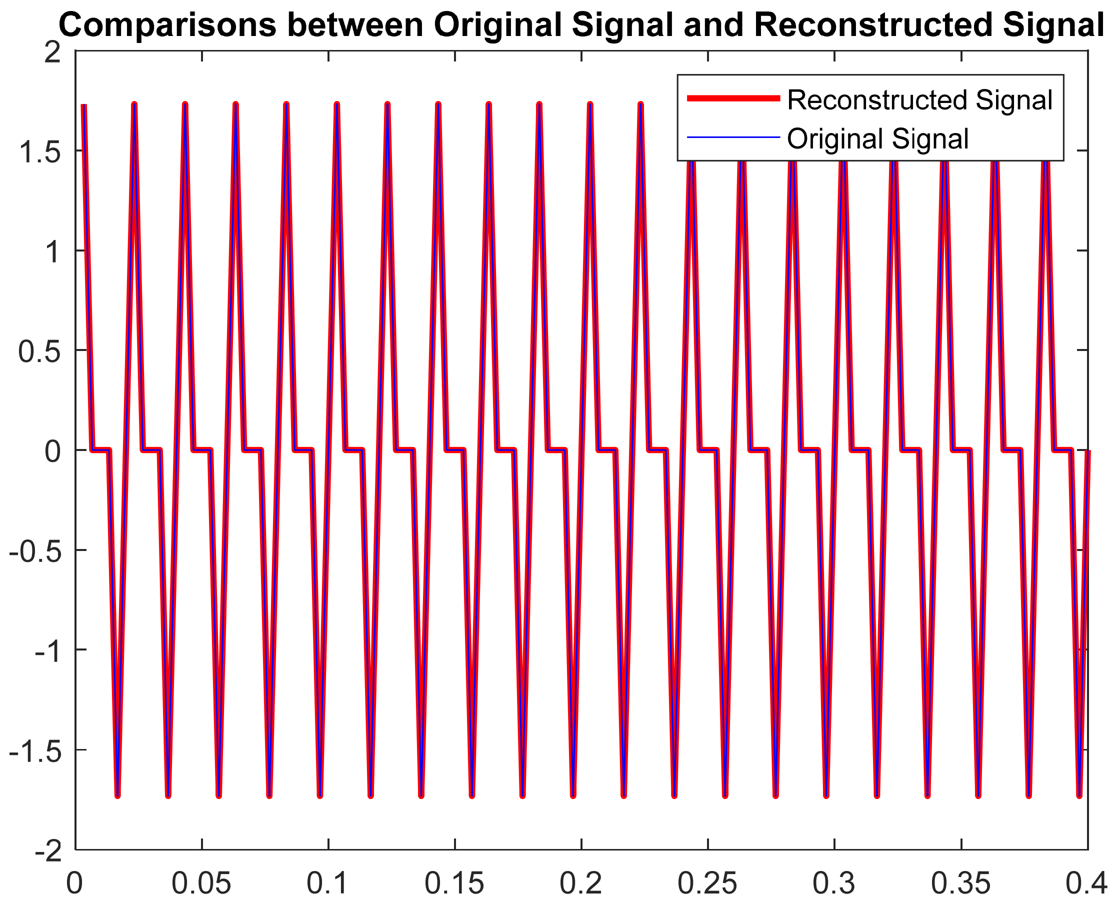

Spectra of the Reconstructed Low-frequency Signal
60
40
20
0
0
50
100
150
200
250
300
Spectra of the Reconstructed High-frequency Signal
60
40
20
0
0
50
100
150
200
250
300

figure6



Comparisons between Original Signal and Reconstructed Signal
Reconstructed Signal
Original Signal
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4

figure7

MATLAB既有函数

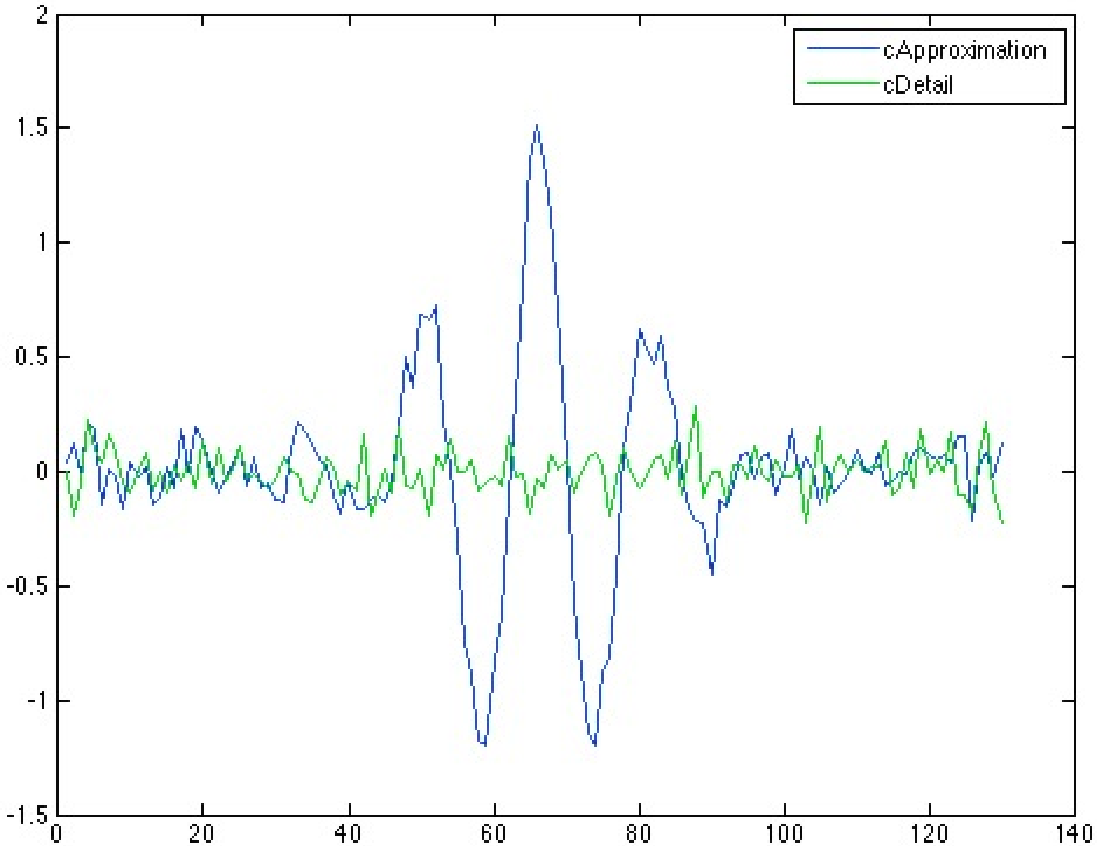
- 函数dwt 对信号进行一层小波分解
- 函数idwt对一层小波分解求逆过程
- 函数wavedec对信号进行N层分解
- 函数waverec对信号进行重建
- 函数swt对信号进行N层小波分解，小波系数与原始数据等长
- 函数iswt对N层swt分解结果进行重构

一层小波分解与重建

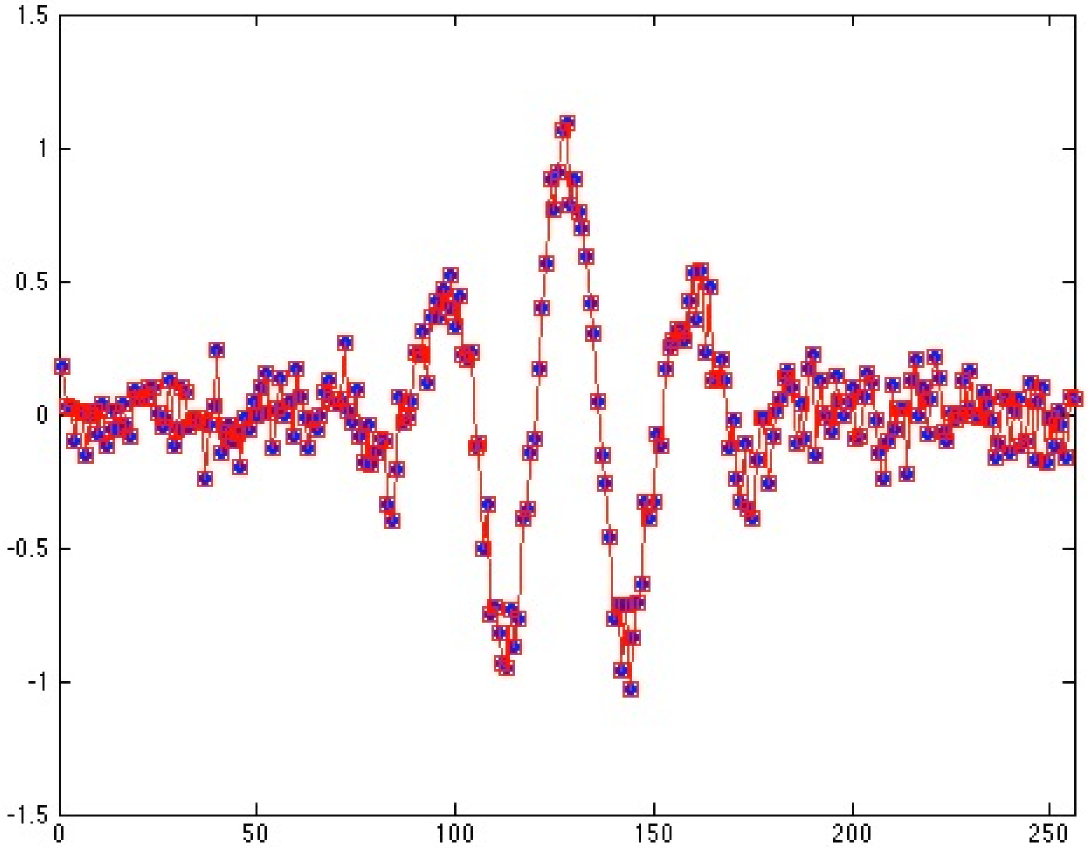
- fs#?#34:#’ #fs = 128Hz#
- t=-1+3/fs:1/fs:1; #
- f?6#
- sinewave#?#cos*4,pi,f,t+#
- w#?#4,*#71*4,pi,f))^2; % the width#parameter#
- gaussian#?#exp((-t.^2)/w#+#
- mwavelet#?#sinewave#0,#gaussian#
- ##
- a#?#randn*size*t++,0B#
- a#?#mwavelet+a#
- ##
- cA#cD#?#dwt*a, ‘db3’); % 一层小波分解#
- X = idwt*cA#cD, 'db3'); % 一层小波重建#

- figure, plot(t, a, '.');#
- hold on; plot(t, X, 'r-');
  #
- figure, plot([cA; cD]');#
- ##
-
  #

一层小波分解结果


cApproximation
cDetail
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
20
40
60
80
100
120
140

原始信号与重建结果



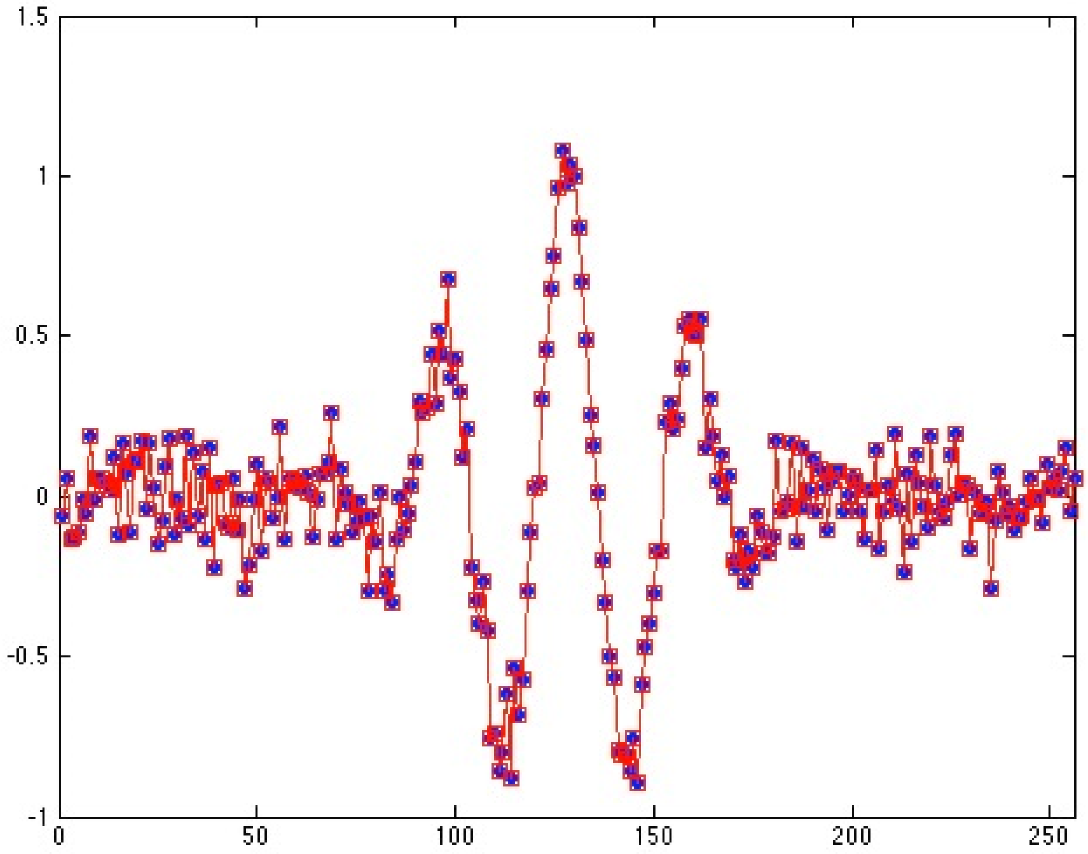
蓝色为原始信号

红色为重建结果

多层小波分解与重建

- [C L] = wavedec*a, 3, 'db3');#
- figure .#plot(C(1:L(1)));#
- hold#on=#plot(C(L(2):L(3)), 'r/)+=#
- hold#on=#plot(C(L(3):L(4)), 'g/)+=#
- ##
- ##
- Y#?#waverec(C, L, 'db3');#
- figure .#plot*a+#
- hold#on=#plot*Y .#)r0)+#
- hold#on=#plot*a .#)bs)+#

原始信号与多层重建结果



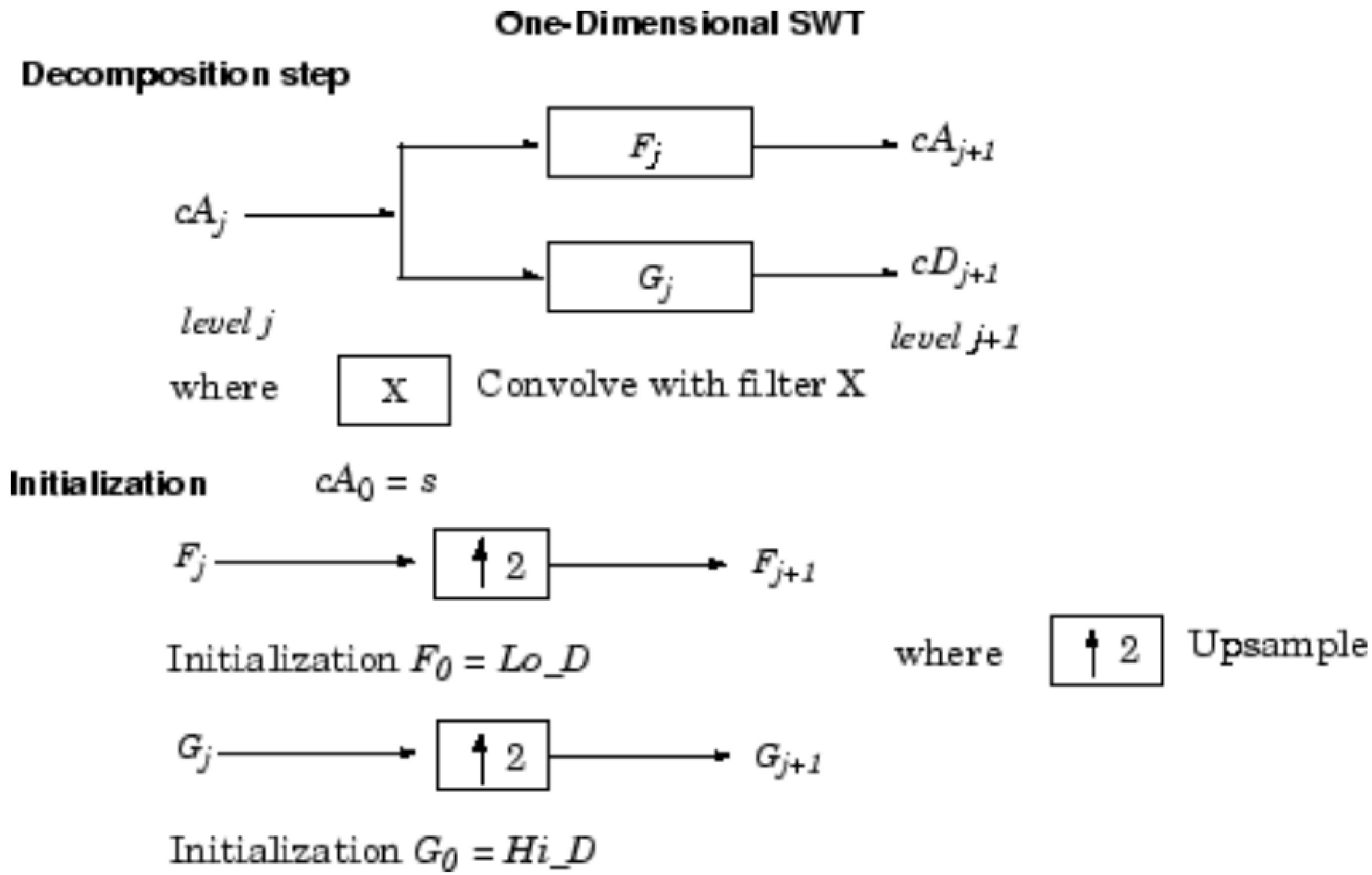
蓝色为原始信号

红色为重建结果

平稳小波变换SWT

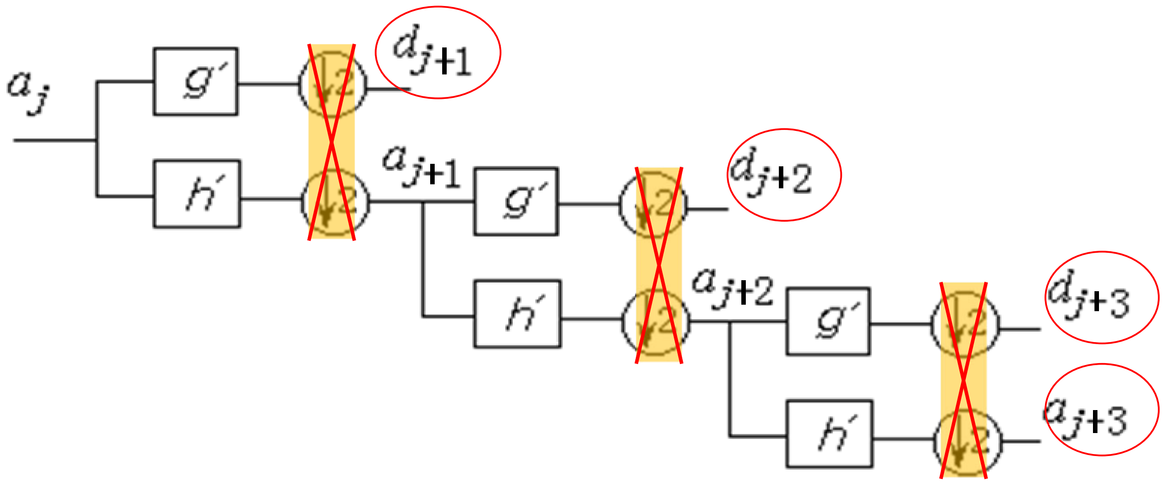
- Stationary Wavelet Transform, SWT
  - MATLAB函数：swt()，iswt()
  - 相对于Mallat关于离散小波变换的快速算法
- 原理：对小波函数进行上采样（upsampling），从而增大小波尺度系数，获取低频部分的逐级细分
- 优点：小波系数与信号具有一致的时间对应关系 —— 时不变特性（time-invariant wavelet representation, Pesquet, J.C.; H. Krim, H. Carfatan 1996)
- 不足：比Mallat算法占用更多内存

逐级分解的swt算法


One-Dimensional SWT
Decomposition step
cA_j
F_j
G_j
cA_{j+1}
cD_{j+1}
level j
level j+1
where
X
Convolve with filter X
Initialization $cA_0 = s$
F_j
2
F_{j+1}
Initialization $F_0 = Lo\_D$
where
2
Upsample
G_j
2
G_{j+1}
Initialization $G_0 = Hi\_D$

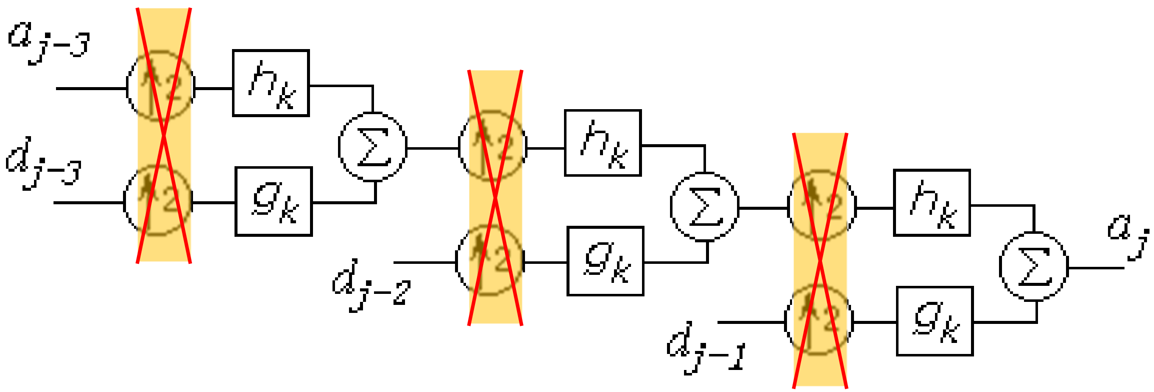
多级分解的swt算法

- 信号分解
  - 相比dwt，小波系数无2倍下采样的过程，保持小波系数与信号的长度一致（时不变）
  - 但是要逐级对滤波器g' h'进行上采样
  - 迭代，实现多级分解



iswt 多级重构

- 信号分解的逆过程
  - 相比dwt，小波系数无2倍上采样的过程，保持小波系数与信号的长度一致（时不变）
  - 但是要逐级对滤波器h_k g_k进行下采样
  - 迭代，实现多级重构



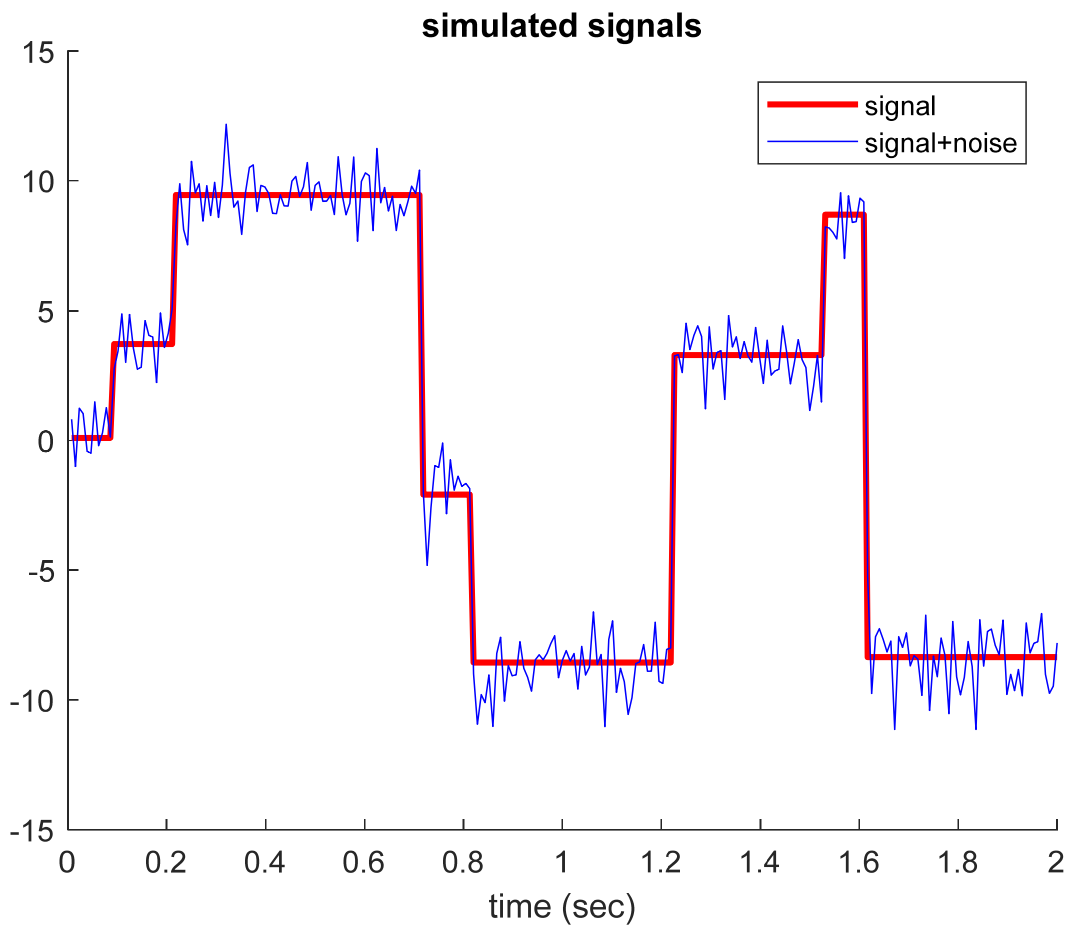
举例：Ex08_swt_vs_dwt.m （信号仿真）

- %% define the parameters
- dataLen = 256; % data points
- sampRate = 128; % Hz
- nSqWaveSegs = round(dataLen/64);
- ampSqWaves = randn(1, nSqWaveSegs)*5;
- tTick = [1:dataLen]/sampRate; % seconds
-
- %% simulate the square waves
- tmp = randperm(dataLen);
- tmp = tmp(1:nSqWaveSegs-1);
- tmp = [sort(tmp) dataLen]; % the indexes of square wave boundaries
- sqWaves = ampSqWaves(1)*ones(1, tmp(1));
- for iSeg = 2:nSqWaveSegs
-     sqWaves = [sqWaves, ampSqWaves(iSeg)*ones(1, tmp(iSeg)-tmp(iSeg-1))];
- end
- sqWaves = sqWaves - mean(sqWaves);
- sqWavesN = sqWaves + randn(1, dataLen);

举例：Ex13_swt_vs_dwt.m （仿真结果）

- figure, hold on;
- plot(tTick, sqWaves, 'r-', 'linewidth', 2);
- plot(tTick, sqWavesN, 'b-');
- xlabel('time (sec)');
- title('simulated signals');
- legend('signal', 'signal+noise');

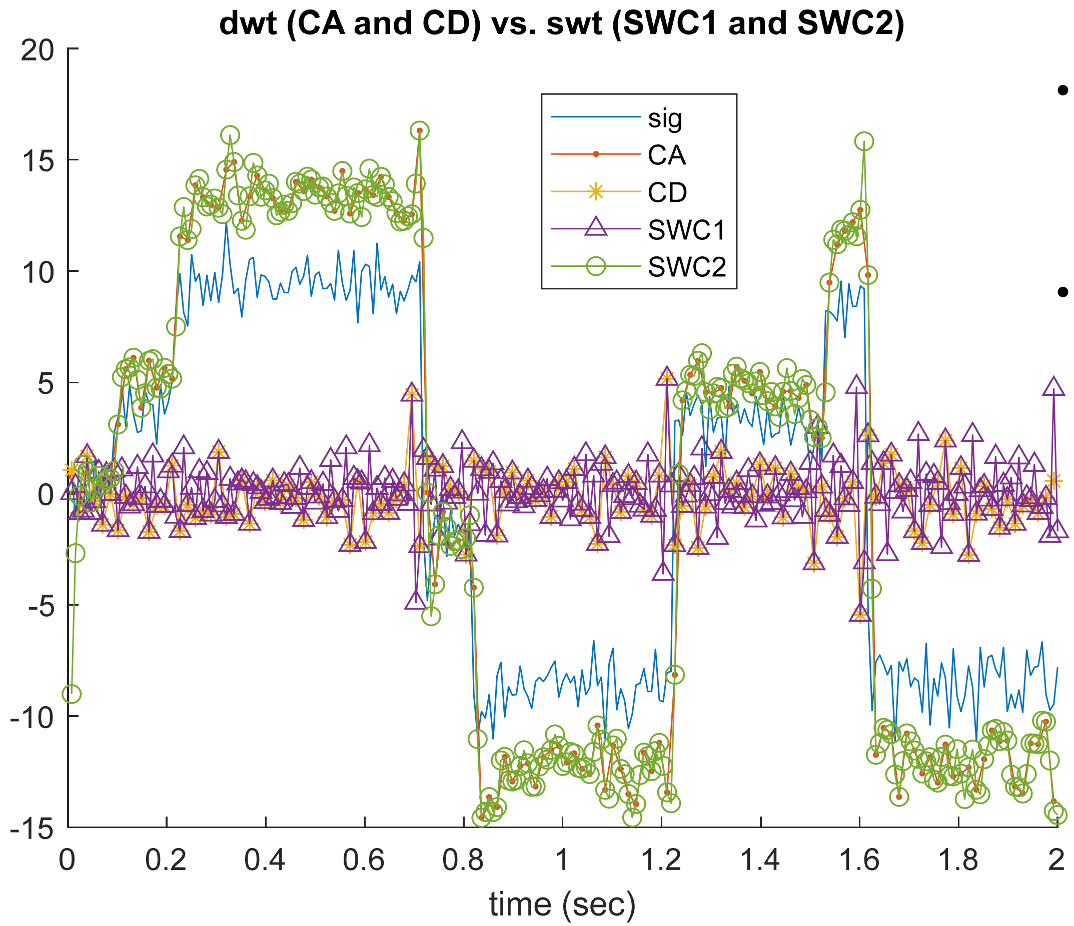
仿真结果，注意每次具体形状是随机的



dwt和swt分解的结果比较

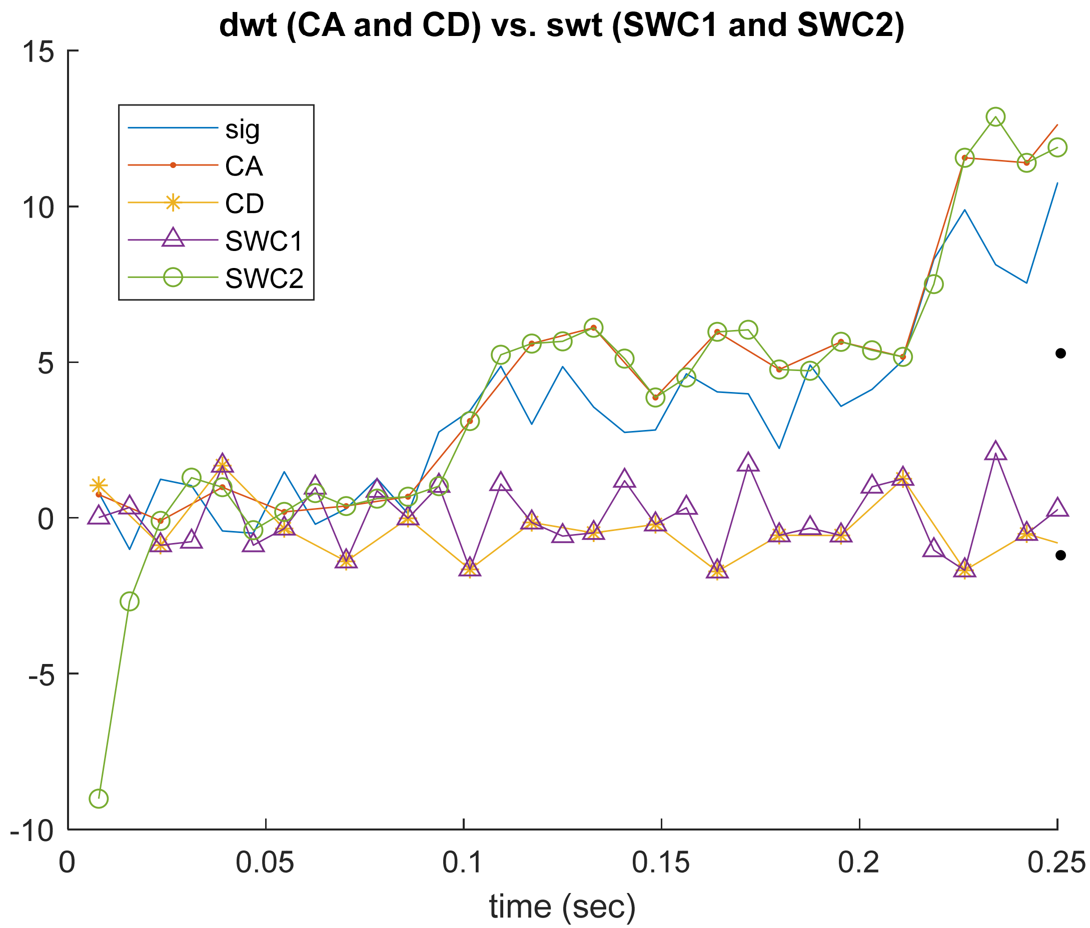
- %% dwt and swt of the target signal
- targetSig = sqWavesN;
- [CA,CD] = dwt(targetSig, 'db3');
- SWC = swt(targetSig, 1, 'db3');
-
- figure, hold on; % show the downsampling of DWT in comparing with SWT coeff's
- plot(tTick, targetSig);
- plot(tTick(1:2:end), CA(2:dataLen/2+1), '.-');
- plot(tTick(1:2:end), CD(2:dataLen/2+1), '*-');
- plot(tTick, SWC(1, :), '^-');
- plot(tTick, SWC(2, :), 'o-');
- title('dwt (CA and CD) vs. swt (SWC1 and SWC2)');
- xlabel('time (sec)');
- legend('sig', 'CA', 'CD', 'SWC1', 'SWC2');

信号的swt与dwt系数的对应关系



- dwt的小波系数是对swt小波系数的基-2下采样的结果
- swt系数与信号等长

细节比较



- dwt的小波系数是对swt小波系数的基-2下采样的结果
- swt系数与信号等长

小结

- 连续小波变换在尺度系数和平移系数选择上的困难
- 离散小波变换的解决方案
- Mallat快速算法的原理
- Harr小波变换举例
- 从频域滤波角度进一步理解离散小波变换
- 离散小波变换相关的MATLAB函数