# ۸

# پیاده سازی سیستمها و روشها

سیستمها را به گونه ای مختلف می توان پیاده سازی خود همانگونه که امکان نوشتن معادلات تصاوی حالت متعدد وجود داشت. آنچه که در انتخاب نوعی از طرح پیاده سازی نقش تعیین کننده ایفا می کند عبارتند از :

- پیچیدگی محاسبات : تعداد ضربها و تعداد جمعها و مقدار مورد نیاز حافظه
- حساسیت طرح به خطا در پارامترهای سیستم ناشی از محدودیت رقمی و محدودیت دقت و محاسباتی

بدون تردید طرح مناسب طرحی است که پیچیدگی محاسباتی کمتر و حساسیت پائینی داشته باشد. با این وجود نوع مسئله و امکانات عواملی هستند که در انتخاب طرح نقش ایفا می کنند. بلادرنگ *real time* یا غیر بلادرنگ بودن عملیات نیز از جمله عوامل بحساب می آیند. عملیات پردازش سیگنال تحت ۳ دسته کلی جای می گیرند

۱- *DFT* بدست آوردن پاسخ فرکانسی سیگنال

۲- کانولوشن : بدست آوردن پاسخ سیستم به ورودی برای این منظور ۳ روش متصور است.

۲-*a* ) محاسبه مستقیم کانولوشن

۲- *b* )براساس معادله دیفرنس و حل ویکوسیو مسئله بر اساس طرحی که مستقیما از معادله دیفرنس و یا تابع تبدیل بدست می آید.

۲- *c* )براساس *IDFT-DFT* که *DFT* ضربه سیستم در *DFT* سیگنال ضرب و از نتیجه *IDFT* گرفته شده تا خروجی بدست آید.

۳- کورلیشن : کورلیشن برای تخمین طیف فرکانسی استفاده می شود و عملیات محاسباتی کورلیشن است لذا راه کارهای کانولوشن برای کورلشن نیز قابل استفاده است.

روش ۲-*a* برای فیلترهای *IIR* که طول *h(n)* نامحدود است قابل استفاده نیست و باید ۲-*b* استفاده شود. برای فیلترهای *FIR* روش های ۲- *a* و ۲- *b* مشابه یکدیگرند. روش ۲- *c* که برای معتبرهای *FIR* قابل استفاده است راندمان محاسباتی بالاتری را از خود نشان می دهد، وقتی که طول ورودی از حدی بزرگتر باشد.

در محاسبه*DFT*، اگر بجای اطلاعات تمام طیف به بخشی از آن نیاز باشد روشی بر اساس ۲-*b*، راندمان بالاتری نسبت به الگوریتمهای محاسبه *DFT* از خود نشان می دهد.

## ۸-۱ طرح پیاده سازی تابع تبدیل سیستمها

آنچه که بین روشها تفاوت می گذارد حساسیت طرح به خطا در پارامترهای سیستم ناشی از محدودیت رقمی است. برای مثال ساختارهای متوالی و موازی و مشبک *Lattice* حساسیت کمتری از خود نشان می دهند.

هدف بحث در این بخش ارائه طرح پیاده سازی سیستمی با تابع تبدیل

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

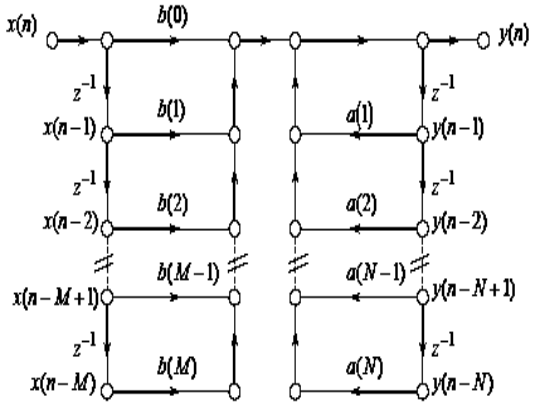


$$H(z)=\frac{Q(z)}{P(z)}=\frac{\sum_{j=0}^{M-1} b_j z^{-j}}{1+\sum_{i=0}^{N-1} a_i z^{-i}}$$

است. در این سیستم اگر $a_k=0$ باشد سیستم *FIR* و در غیر اینصورت *LIR* است.

### ۸-۱-۱ روش ۱ مستقیم (*nonCanomic* غیرپیراسته)

آنچه که بطور مستقیم از تابع تبدیل سیستم می توان نتیجه گرفت طرح شکل است. که یک سیستم تمام صفر سری با یک سیستم تمام قطب قرار گرفته اند.




$$H(z)=\frac{Q(z)}{P(z)}=Q(z)\frac{1}{P(z)}=\sum_{j=0}^{M-1} b_j z^{-j} * \frac{1}{1+\sum_{i=0}^{N-1} a_i z^{-i}}$$

این طرح نیازمند: *N+M+1* ضرب

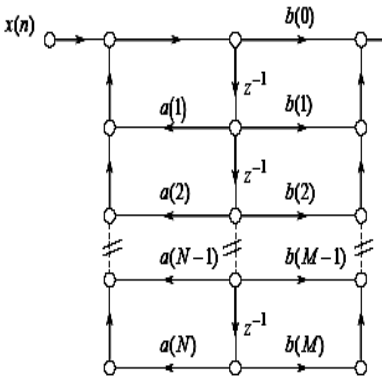
*N+M* جمع

*N+M+1* خانه حافظه است

این طرح از نظر ساختاری حداقل (می نی مال *minimal*) نیست و استفاده نمی شوند.

### ۸-۱-۲ روش۲ مستقیم (*Cononic* پیراسته)

**سیستم *IIR*:** اگر در نوشتن *H(Z)* تغییر ایجاد کنیم و بصورت




$$H(z)=\frac{1}{P(z)}Q(z)$$

بنویسیم. روش ۲ مستقیم مطابق شکل بدست می آید.

این طرح نیازمند: *N+M+1* ضرب

*N+M* جمع

*max{N,M}* خانه حافظه است

از این ساختار در مدل فضای حالت استفاده شد.

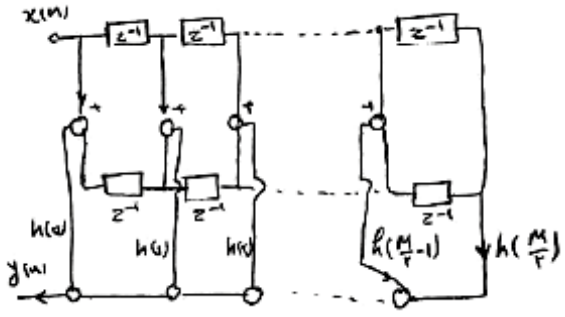
**سیستم *FIR*** درفیلتر *FIR* ضرایب $a_k$ صفر هستند. بنابراین برای *FIR* روش ۱ و ۲ مستقیم تفاوتی با یکدیگر ندارند.

این طرح نیازمند: *M* ضرب

*M* جمع

*M-1* خانه حافظه است

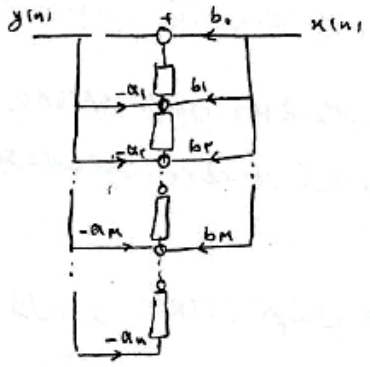
**سیستم *LPFIR*** : در این سیستمها برای پیاده سازی طرح از تفارن موجود در پاسخ ضربه سیستم استفاده می شود که *h(n)=±h(M-n) r=0,…., M* :




به این ترتیب تعداد ضرب از *M* به حدود *M/2* کاهش می یابد(بستگی به زوج یا فرد بودن *M* و نوع فیلتر این عدد ممکن است*M/2*، *M/2+1*، یا *(M+1)/2*گردد)

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب





### ۸-۱-۳ مدل ترانسپز(Transpose) ۲ مستقیم

قضیه : اگر در بلوک دیاگرامی جهت ها معکوس و جای ورودی با خروجی عوض شود تابع تبدیل مدل تغییر نمی کند. شکل مدل ترانسپز ۲ مستقیم را نشان می دهد از این مدل نیز در فضای حالت ب (نوع دوم) استفاده شد.

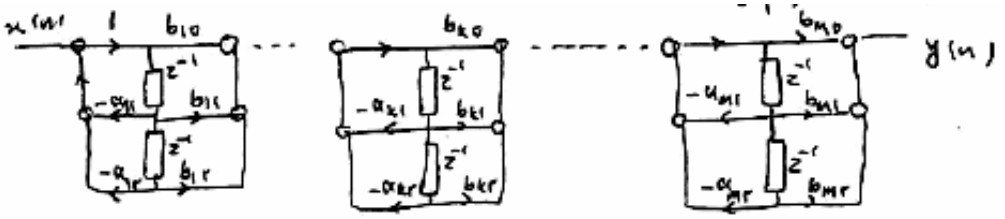
### ۸-۱-۴ مدل متوالی

**سیستم *IIR*** : با محاسبه صفرها و قطبهای $H(z)$ آنرا می توان بصورت حاصلضرب جملات کسری درجه ۲ نوشت.

$$H(z)=b\prod_{1}^{M}\frac{1+b_{k1}z^{-1}+b_{k2}z^{-2}}{1+a_{k1}z^{-1}+a_{k2}z^{-2}}$$

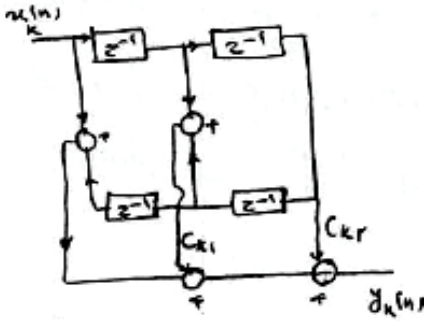
اگر تعداد صفرها فرد باشد یک جمله درجه ۱ نیز بدست می آید. بهره $b$ را می باید به نحوی بین زیر مدلها تقسیم نمود.

حالا می توان جملات کسری را با روش ۲ مستقیم پیاده سازی نمود. تعداد ضربها و تقسیم ها مشابه حالت ۲ است.



**سیستم *FIR*** در سیستم *FIR* صرفا ضرایب $a_k$ صفر می گردند و تعداد محاسبات با روش ۲ مستقیم *FIR* تغییر نمی کند.

**سیستم *LPFIR*** : در حالت *FIR* خطی فاز مناسب بجای سری جملات درجه ۲ جملات درجه ۴ سری شوند تا از تقارن در پاسخ ضربه استفاده گردد. این امر موجب کاهش ۵۰ درصدی تعداد ضربها می گردد. تا با طرح *LPFIR* به روش ۲ مستقیم مطابقت یابد. شکل این ساختار را نشان می دهد.



بستگی به نوع فیلتر خطی فاز ممکن است تغییراتی از نظر علامت در شکل ضرورت پیدا کنند.

دستور *MATLAB* تابع تبدیل $H(Z)$ را به حاصلضرب جملات درجه ۲ استخراج می کند. *sos* ضرایب جملات درجه ۲ را جداگانه در یک ماتریس ارائه می کند.

```
[z, p, k]=tf2zp(num,den); sos=zp2sos(z,p,k);
```

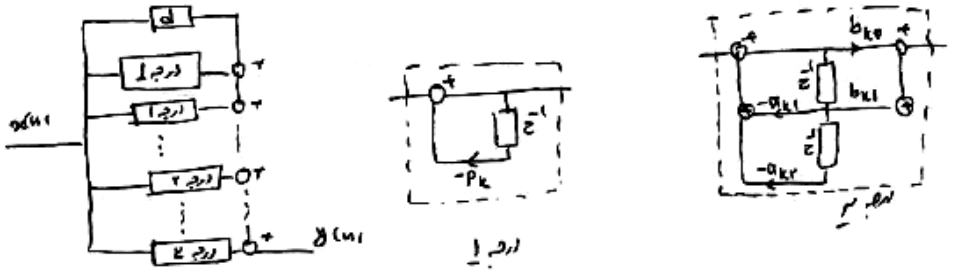
### ۸-۱-۵ مدل موازی ( فقط *IIR*)

تابع تبدیل $H(z)$ را می توان با بسط به کسرهای جزئی آنرا بصورت مجموع کسرهای درجه ۱ یا ۲ نوشت.

$$H(z)=d+\sum_{k}\frac{A_k}{1-p_k z^{-1}}+\sum_{k}\frac{b_{k0}+b_{k1}z^{-1}}{1+a_{k1}z^{-1}+a_{k2}z^{-2}}$$

اگر $M>N$ باشد ابتدا با تقسیم صورت به مخرج یک چند جمله ای غیرکسری به اضافه چند جمله ای کسری بدست می آید. بلوک دیاگرام نتیجه این تجربه مطابق شکل خواهد گردید.

www.MatlabKar.com

وقتی تابع تبدیلی اینگونه تجربه شود معادله فضای حالت شکل قطری به خود می گیرد.

$$v(n+1)=\begin{bmatrix} P_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & P_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & P_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -a_{12} & -a_{11} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -a_{22} & -a_{21} \end{bmatrix} v(n)+\begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} x(n)$$

$$y(n)=\begin{bmatrix} A_1 & A_2 & A_3 & (-b_0 a_{12}) & (b_{12}-b_1 a_{12}) & (-b_{22} a_{22}) & (b_{23}-b_1 a_{22}) \end{bmatrix} v(n)+d\,x(n)$$

در سیستم مثال فوق ۳ قطب ساده $P_1$ , $P_2$ و $P_3$ و دو قطب کمپلکس نشان داده شده اند.

دستور *residuez* , *residue* تابع تبدیل را به مجموع کسرهای درجه ۱ تجزیه می کند.

## ۸-۱-۶ نمونه گیری از پاسخ فرکانسی (فقط *FIR*)

تعداد عملیات ضرب و جمع به تعداد قابل ملاحظه ای کاهش می یابد اگر بجای نوشتن تابع تبدیل *FIR* با ضرایب $h(n)$ آنرا با ضـرایبی از $H(k)$ بنویسیم. ذیلا عملیات شرح داده می شود.

۱- نمونه برداری از $H(w)$ در فرکانس‌های $W_k$

$$W_k=\frac{2\pi}{M}(k+\alpha) \quad k=0,\dots,\frac{M-1}{2} \text{ زوج } M$$

$$k=0,\dots,\frac{M}{2}-1 \text{ فرد } M$$

$\alpha=0$ اگر $H(0)\neq 0$

$\alpha=\frac{1}{2}$ اگر $H(0)=0$

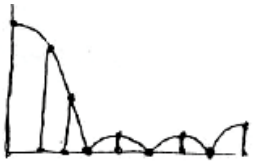
$$H(k+\alpha)=\sum_{n=0}^{M-1} h(n)\,e^{-j2\pi(k+\alpha)n/M} \qquad k=0,\dots,M-1$$

۲- بدست آوردن $H(z)$ از $H(k+\alpha)$

$$h(n)=\frac{1}{M}\sum_{k=0}^{M-1} H(k+\alpha)\,e^{j2\pi(k+\alpha)n/M}$$

$$\Rightarrow H(z)=\sum_{n=0}^{M-1} h(n)z^{-n}=\frac{1-z^{-M}e^{j2\pi\alpha}}{M}\sum_{k=0}^{M-1}\frac{H(k+\alpha)}{1-e^{j2\pi(k+\alpha)/M}z^{-1}}$$

۳)پیاده سازی $H(Z)$ که حالا ضرایب آنرا $H(k+\alpha)$ می سازند.



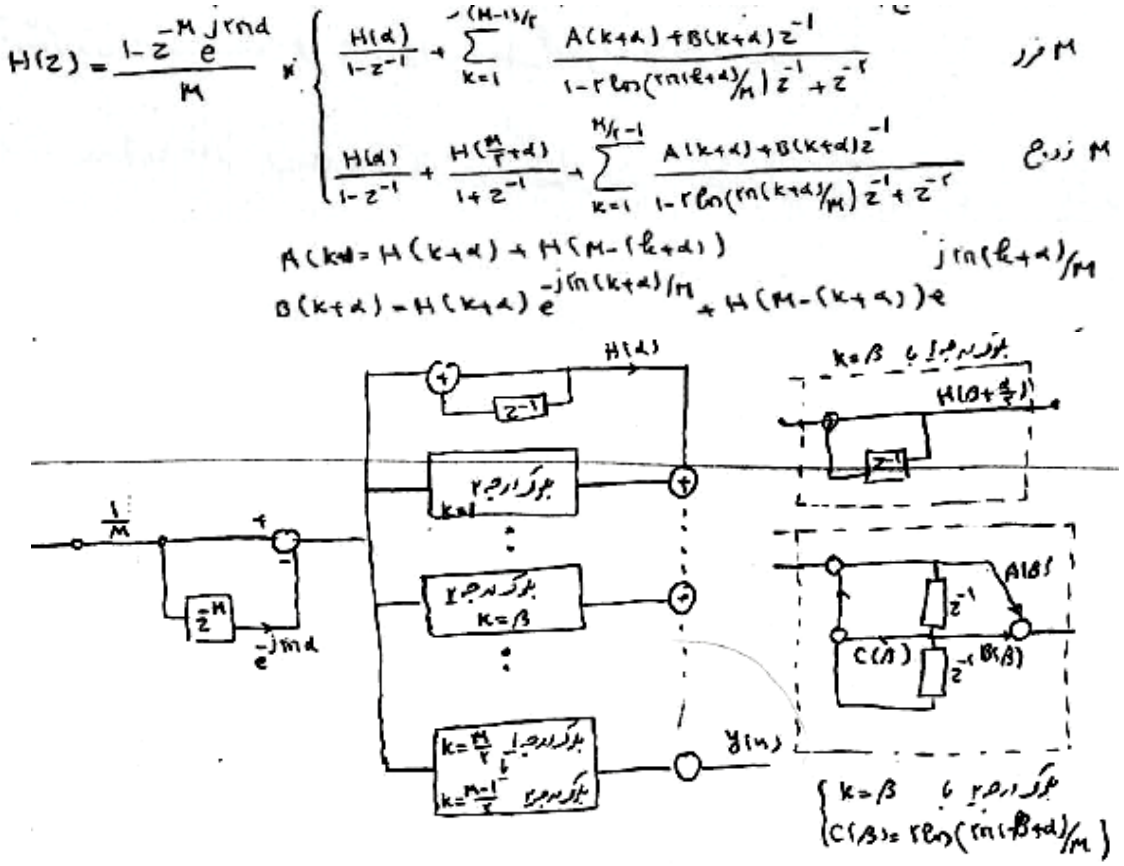
۴)مزیت از آنجا ناشی می شود که $H(k+\alpha)$ در تعـدادی نقـاط صـفر هـستند در حالیکـه $h(n)$ اینگونه معلوم نیست باشد از اینرو تعدادی از ضربها و جمعها حذف شده و تعداد عملیات کـاهش می یابد. برای مثال به $H(\omega)$ شکل دقت کنید از ۹ تعداد $H(k)$ تعداد۳ عدد صفر و فقط ۶ عدد دارای مقدار است.

۵)بهره گیری از تقارن در $H(k)$: می دانیم $H(h+\alpha)=H^{+}(M-(k+\alpha))$ است اضافه کردن این تقارن به رابطه کاهش بیـشتر ضـرب و

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

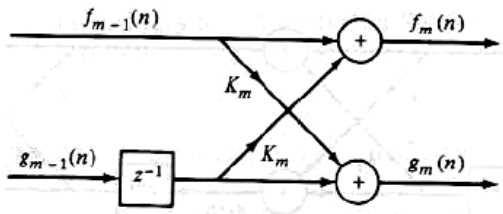


جمعها را به دنبال دارد. بستگی به فرد یا زوج بودن $M$ نتایج با اندکی تفاوت بدست می آید.



در یک مثال، روش مستقیم۲ به ۳۲ ضرب نیاز داشت که به دلیل تقارن در $h(n)$ به ۱۶ کاهش می یافت. همان سیستم به روش نمونه گیری پاسخ فرکانسی تنها به ۹ ضرب و ۱۳ جمع نیاز پیدا کرد.

### ۸-۱-۷ طرح مشبک *Lattice*



در طرح مشبک تابع تبدیل بصورت اتصال متوالی طبقات درجه ۱ مشبک پیاده می شود. شکل یک طبقه مشبک را نشان می دهد. در این طرح به $k_1$ ضرب انعکاسی می گویند.

رابطه بین ورودی و خروجی این شبکه از اینقرار است.

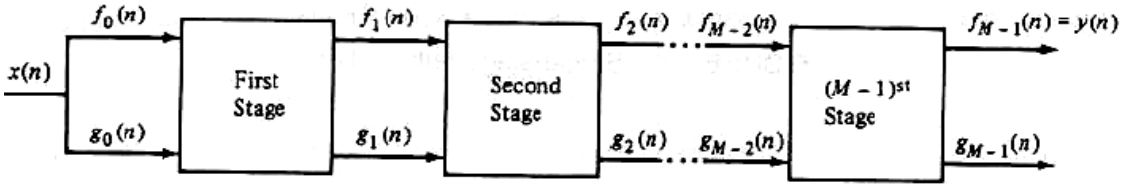
$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1)$$
$$g_m(n) = K_m f_{m-1}(n) + g_{m-1}(n-1)$$

#### سیستم *FIR*

سیستم *FIR* با تابع تبدیل

$$H(z) = \sum_{k=0}^{M-1} b_k z^{-k} = b_0 \left(1 + \sum_{k=1}^{M-1} b'_k z^{-k}\right)$$

را می توان با $M$-1 طبقه طرح مشبک درجه ۱ ساخت



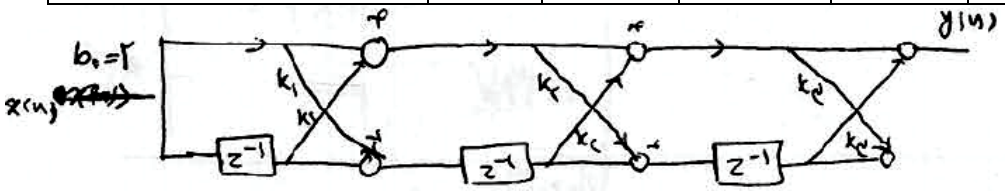
www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



هر طبقه دارای ضریب انعکاسی $k$ است که محاسبه $k_1$ , $k_2$ تا $k_{M-1}$ همان ضرایبی است که روش شور- کوهن برای بررسی پایداری ارائه می کند.

مثال : برای سیستم $y(n)=2x(n)+4x(n-1)+2x(n-2)+6x(n-3)\Rightarrow H(z)=2(1+2z^{-1}+z^{-2}+3z^{-3})$ طرح مشبک بدست آورید.

| | | | | | | |
|---|---|---|---|---|---|---|
| ضرایب چند جمله ای با $a0=1$ | $a3$ | 1 | 2 | 1 | 3 | **k3** |
| معکوس ضرایب چند جمله ای | $a'3$ | 3 | 1 | 2 | 1 | |
| $a_2=\frac{a_3-k_3a'_3}{1-k_3^2}$ | $a2$ | 1 | 0.125 | 0.625 | **k2** | |
| | $a'2$ | 0.625 | 0.125 | 1 | | |
| $a_1=\frac{a_2-k_2a'_2}{1-k_2^2}$ | $a1$ | 1 | 0.077 | **k1** | | |
| | $a'1$ | 0.077 | ۱ | | | |




از آنجائیکه $|k3|=3>1$ است به این معنی است که $H(z)$ دارای صفری خارج از دیسک واحد است.

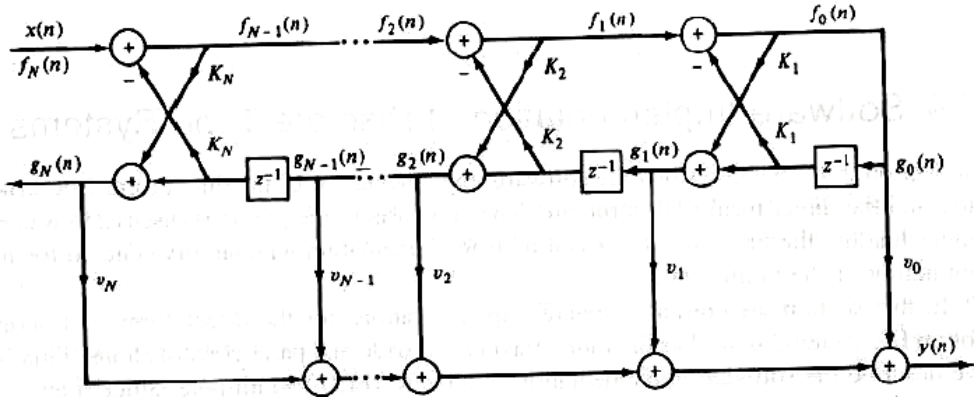
دستور $k=poly2rc(num)$ فیلتر $FIR$ را به مشبک تبدیل کرده و $k$ ضرایب را می دهد.

**شبکه مشبک نردبانی برای فیلتر *IIR***

ساخت فیلتر $IIR$

$$H(z)=\frac{\sum_{j=0}^{M-1}b_jz^{-j}}{1+\sum_{i=0}^{N-1}a_iz^{-i}}$$

از دو قسمت مشبک تمام قطب و نردبانی برای صفرها مطابق شکل ساخته می شود.



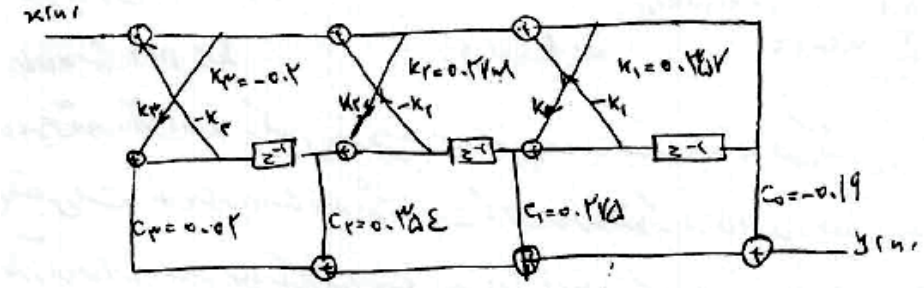

ضرایب انعکاسی $k$ را مخرج کسر و ضرایب $c=v$ را صورت کسر تعیین می کنند. نحوه محاسبه $k$ , $c$ را مثال ذیل روشن می دهد.

$$H(z)=\frac{0.433z^{-1}+0.346z^{-2}+0.054z^{-3}}{1+0.154z^{-1}+0.077z^{-2}+0.052z^{-3}}$$

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ضرایب چند جمله ای | a3 | 1 | 0.4 | 0.18 | **k3= -0.2** | 0 | 0.44 | 0.362 | **c3= 0.02** | b3 |
| معکوس ضرایب چند جمله ای | a'3 | -0.2 | 0.18 | 0.4 | 1 | -0.2 | 0.18 | 0.4 | 1 | a'3 |
| $a_2 = \frac{a_3 - k_3 a'_3}{1 - k_3^2}$ | a2 | 1 | 0.454 | **k2= 0.2708** | 0.004 | 0.436 | **c2= 0.354** | | | b2=b3-c3a'3 |
| | a'2 | 0.270 | 0.454 | 1 | 0.270 | 0.454 | 1 | | | a'2 |
| $a_1 = \frac{a_2 - k_2 a'_2}{1 - k_2^2}$ | a1 | 1 | **k1= 0.357** | | -0.091 | **c1= 0.275** | | | | b1=b2-c2a'2 |
| | a'1 | 0.357 | 1 | | 0.357 | 1 | | | | a'1 |
| | | 1 | | | **c0= -0.19** | | | | | |



طرح مشبک نسبت به طرحهای دیگر نیاز به تعداد ضرب و جمع اندکی بیشتر دارد ولی حساسیت کمتری نسبت به خطای مقدار پارامترهای ناشی از محدودیت ولتی و محدودیت محاسباتی از خود نشان می دهد.

دستور *poly2zrc* فیلتر تمام گذر *H(z)* را به طبقات متوالی مشبک تجزیه و ضرایب را تولید می کند. و دستور *tf3latc* تابع تبدیل را به طبقات مشبک متوالی تجزیه می کند.

***{k,c]=tf2latc(num,den}; [num,den]=latc2tf(k,c)***

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



## ۸-۲ طرح پیاده سازی DFT

برای محاسبه DFT رابطه

$$X[k]=\sum_{n=0}^{N-1}x[n]e^{-j\frac{2\pi kn}{N}} \qquad 0\le k\le N-1$$

باید حل گردد. اگر x(n) را کمپلکس در نظر بگیریم انجام مستقیم محاسبات نیازمند

- $N^2$ ضرب کمپلکس و
- N(N-1) جمع کمپلکس است.
- علاوه بر اینها محاسبه توابع مثلثاتی است که چون می تواند فقط یک بار انجام شده و در جدولی نگهداری می شود جزء حجم محاسبات منظور نگردیده است.
- عملیات مربوط به آدرس کردن حافظه برای بیرون آوردن اطلاعات نیز به حساب نیامده است.

هدف بحث این بخش شرح الگوریتمهایی است که می تواند حجم محاسبات را کاهش دهند به این کلاس الگوریتمها( Fast FFT ). Foonier Tr می گویند. در محاسبه مستقیم DFT از تقارن و پریودیک بودن تابع اکسپونانسیل کمپلکس استفاده نشده است استفاده از این مزیت تعداد محاسبات را به $N\log_2 N$ برای ضرب و $2N\log_2 N$ جمع کمپلکس کاهش می دهد. محاسبه DFT ممکن است برای ۳ نوع استفاده مورد نیاز باشد.

۱- تمام طیف نیازی نباشد و فقط بخشی از آن مورد نظر باشد: الگوریتم مناسب در این حال FFT نیست و روشهای مستقیم ترجیح داده می شوند.

۲- تمام طیف مورد نیاز است : تنها انتخاب الگوریتمهایی با راندمان FFT هستند

۳- بدست آوردن H(z) روی کانتوری به غیر از دایره واحد مورد نظر است : برای این منظور تبدیل chirpz مورد استفاده قرار می گیرد که محاسبات آن بر اساس الگوریتمهایی FFT است.

### ۸-۲-۱ الگوریتم گورتزل Goertzel's Algorithm

۱- الگوریتم گورتزل، محاسبه DFT را به روش کانولوشن انجام می دهد.

۲- برای محاسبه یک نقطه DFT، به $(N+2)/2$ ضرب کمپلکس و N+2 جمع کمپلکس نیاز است که تعداد ضربها تقریبا نصف محاسبه مستقیم DFT است. بنابر این حجم محاسباتی آن اندکی کمتر از روش مستقیم است ولی هنوز برای N-DFT تقریبا $N^2$ عملیات ضرب و جمع کمپلکس نیاز دارد.

۳- اگر تعداد نقاط مورد نظر از طیف H(k)، $M\le \log_2 N$ باشد، این روش بر FFT که تمام نقاط را محاسبه می کند ترجیح دارد.

نحوه محاسبه DFT سیگنال x(n) به روش گورتزل از اینقرار است که DFT در $W_N^{-Nk}=1$ ضرب می گردد.

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{nk}=\sum_{n=0}^{N-1}x(n)W_N^{nk}W_N^{-Nk}=\sum_{n=0}^{N-1}x(n)W_N^{(N-n)(-k)}=x(n)\otimes W_N^{-kn}\Big|_{n=N}$$

در نهایت محاسبه DFT بصورت کانولوشن x(n) و $W_N^{-kn}$ بدست می آید که در مقدار n=N محاسبه شود. رابطه کانولوشن بدست آمده به این معنی است که دنباله x(n) توسط سیستمی با پاسخ ضربه $h_k(n)=W_N^{kn}$ فیلتر می شود و خروجی $y_k(n)$ به ازای n=N مقدار X(k) است.

تابع تبدیل این چنین سیستمی از اینقرار است:

$$H(k)=\sum_{n=0}^{N-1}h(n)z^{-1}=\sum_{n=0}^{N-1}W_N^{nk}z^{-1}=\frac{1}{1-W_N^{-k}z^{-1}}$$

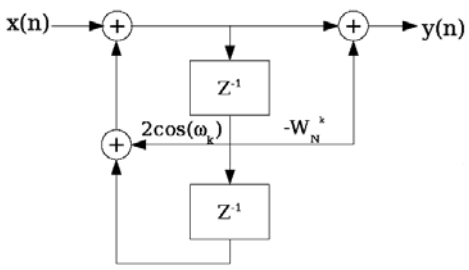
که معادله تفاضلی آن برابر است با: $y_k(-1)=0$ $y_c(n)=W_N^{-k}y(n-1)+x(n)$

این معادله تفاضلی ضریب کمپلکس دارد، برای کاهش محاسبات کمپلکس درجه سیستم را با افزودن قطب مزدوج به آن یک درجه بالا برده تا یک سیستم جدیدی بدست آید.

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



$$H_k(z) = \frac{1 - W_N^k z^{-1}}{1 - 2\cos(2\pi k/N) z^{-1} + z^{-2}}$$




در سیستم اول برای محاسبه هر $y_k(n)$ به ۴ ضرب و ۴ جمع حقیقی نیاز بوده ولی در طرح دوم، اگر ورودی کمپلکس باشد، به ۲ ضرب و ۴ جمع حقیقی نیاز است که کاهش ۵۰ درصدی تعداد ضربها را نشان می دهد. از آنجائیکه هدف بدست آوردن $H(k)=y_k(N)$ است، ضرب $W_N^k$ فقط یک بار انجام می گردد. برای این منظور بجای $y_k(n)$ فقط $V_k(n)$ محاسبه می شود و در $n=N$, $y_k(N)=V_k(N)+W_N^k V_k(N)$ محاسبه می گردد. اگر ورودی کمپلکس باشد تعداد محاسبات برای محاسبه هر $H(k)$ برابر $2(N+2)$ ضرب حقیقی و $4(N+1)$ جمع حقیقی است که این مقادیر نصف تعداد مورد نیاز روش مستقیم است و برای هر ورودی حقیقی نیز خود نصف می گردد. این روش وقتی تعداد نقاط مورد نظر $H(k)$ کمتر از $\log_2 N$ باشد بر روشهای با راندمان دیگر ترجیح دارد.

## ۲-۲-۸ *Fast Fourier Transform (FFT)*

مجموعه الگوریتمهائی با راندمان محاسباتی قابل ملاحظه هستند که برای محاسبه *DFT* مورد استفاده قرار می گیرند. این الگوریتمها از تقارن و پریودیک بودن $H(k)$ کمال استفاده را در محاسبه *DFT* می برند. گوس بعنوان اولین ارائه دهنده *FFT (1805)* شناخته می شود. بعد از آن در خلال سالهای ۱۸۰۵ تا ۱۹۶۵، چندین روش مستقل ارائه شده است تا اینکه کولی و توکی ***Cooley & Tukey (1965)*** اولین الگوریتم *FFT* که شهرت عمومی پیدا کرد و تحولی در پردازش دیجیتالی ایجاد کرد را ارائه کردند. .

ولی وجه تمایز آنها با یکدیگر الگوریتمهائی که آنها نیز تقارن و پریودیک بودن $H(k)$ را لحاظ می کند در موضوع ذیل نهفته است.

۱- محاسبه مستقیم *DFT* با $N$ نقطه نیازمند $N^2$ ضرب و $N(N-1)$ جمع کمپلکس است. فرض شده که $W_N^{kn}$ قبلا محاسبه شده اند.

۲- اما اگر $N$ نقطه *DFT* از طریق $L$ تا *DFT* با $M$ نقطه صورت گیرد که $N=LM$ باشد تعداد ضرب و جمعها کاهش می یابد. $L$ تا *DFT* با $M$ نقطه + عملیاتی میانی $+M$ تا *DFT* با $L$ نقطه کلا به $N(M+L+1)$ ضرب و $N(M+L-2)$ جمع کمپلکس نیازمند است.

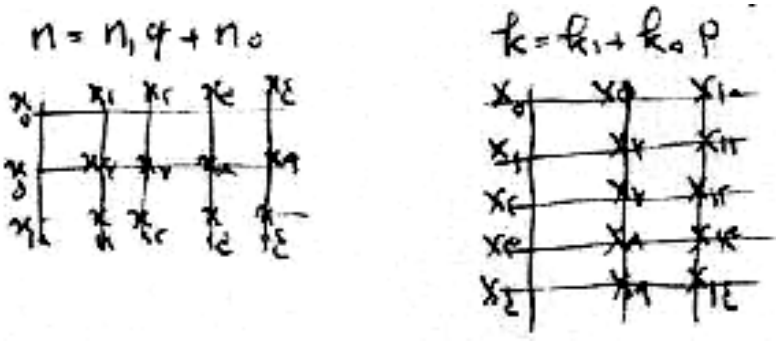
- برای مثال اگر $L=N/2$ در نظر گرفته کاهش ۵۰درصدی محاسبات را در پی دارد.
- به همین ترتیب اگر $N$ بصورت متوالی تقسیم شود، $N=r_1 r_2 r_3 r_4 r_5 \ldots$ از این مزیت بخوبی می توان استفاده کرد.
- این مزیتها زمانی وجود دارد که محاسبه کامل طیف مدنظر باشد در غیراینصورت راندمان موجود در *FFT* با محاسبه نقاط غیر ضرور به هدر می رود.

### تجزیه یک گام *DFT* به ۲ گام

برای روشن شدن موضوع یک دنباله ۱۵تایی به ۳ دنباله ۵ تایی تجزیه و با بکارگیری پریودیسیتی در طیف ، کاهش محاسبات نشان داده می شود. به این ترتیب *15-DFT* را با بلوکهای *3-DFT* و *5-DFT* محاسبه می نماییم.

دنباله ۱۵تایی به ۵ دنباله ۳ تایی تجزیه می شود. $N=15=3*5=p*q \Rightarrow p=3,\ q=5$

با این تجزیه دنباله ورودی $x(n)$ و خروجی $X(k)$ بصورت ماتریسهای شکل چیده می گردند.

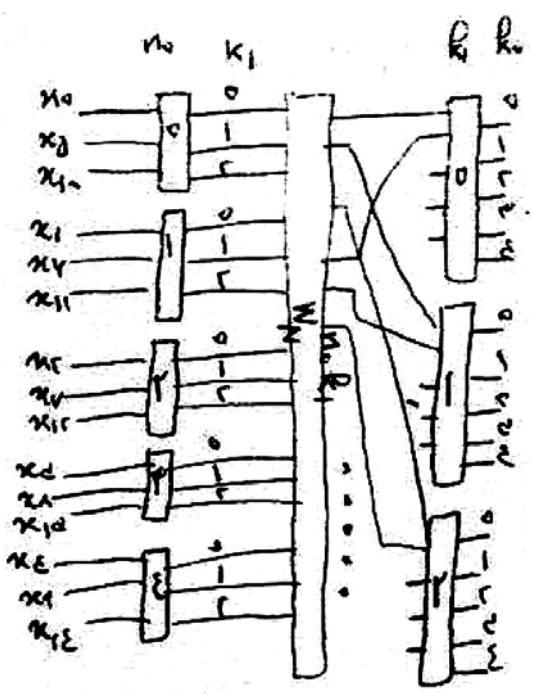


www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



بر این اساس رابطه *DFT* نوشته می شود.

$$X(k)=\sum_{n} x(n)W_N^{kn}=\sum_{n_1,n_0} x(n_1,n_0)W_N^{kn}=\sum_{n_1,n_0} x(n_1,n_0)W_N^{(n_1q+n_0)(k_1+k_0p)}=$$



باتوجه به

$$W_N^{n_1qk_1}=e^{-j\frac{2\pi}{N}n_1qk_1\frac{P}{P}}=W_P^{n_1k_1}$$

که پریودیسیتی در محاسبه *DFT* را می رساند می نویسیم

$$\sum_{n_1,n_0} x(n_1,n_0)W_N^{n_1qk_1}W_N^{n_1qk_0p}W_N^{n_0k_1}W_N^{n_0k_0p}=$$

$$\sum_{n_1,n_0} x(n_1,n_0)W_P^{n_1k_1}W_N^{n_0k_1}W_N^{n_0k_0}=$$

$$\sum_{n_0=0}^{q-1}\left[W_N^{n_0k_1}\left[\sum_{n_1=0}^{p-1}x(n_1,n_0)W_p^{n_1k_1}\right]\right]W_N^{n_0k_0}$$

رابطه اخیر نشان می دهد که *15-DFT* بـه ۵ تـا *3-DFT*، عملیـات ضـرب میانی و در نهایت ۳ تا *5-DFT* تجزیه شده است. شکل این عملیات را نشان می دهد.

در طرح *15-DFT* تعداد ضربها $۱۵^۲=۲۲۵$ ولی در طرح فوق این تعداد به

$5*3^2+15+3*5^2=135$

کاهش یافته است.

- آرایش دنباله *x* را به هر صورتی می توان بصورت ماتریس نوشت ولی طرح فوق ، طرح *decimation in frequency* است. در این طرح بردار ورودی *x* مرتب ولی بردار خروجی *X* نامرتب *bit reversed* است. در این طرح گامهای میانی نیـاز بـه حافظـه جداگانـه ندارند.
- اگر قراردادن ورودی و خروجی در ماتریسها تغییر می کرد، روش *decimation in time* بدست مـی آمـد ه در آن *x* نامرتـب *bit reversed* ولی خروجی مرتب بود. برای این امر *k=k1p+k0* و *n=n1+n0q* باید تعریف گردد
- البته تجزیه به گروههای ۳ و ۵ تایی انتخاب مناسبی نیست، ولی فقط به هدف شرح مسئله مورد استفاده قرار گرفت اگر به مجموعه ورودی صفری اضافه می شد تا طول آن به ۱۶ برسد و سپس در ۴ گام تجزیه می شد *(16=2*2*2*2)*، ۴طبقه عملیات صورت مـی گرفت که در هر طبقه ۴ عملیات *2-DFT* انجام می گشت.در این حال تعداد محاسبات به *N/2 log₂N=32* کاهش می یابد.

## ۱: مبنای ۲

در این الگوریتمها $N=2^\nu$ تجزیه می گردد اگر *N* برابر $۲^\nu$ نباشد با افزودن صفر به آن شرایط برقراری تساوی فراهم می شود. در این روش تعداد ضربه ها ی کمپلکس به $N/2\ log_2N$ و تعداد جمعهای کمپلکس به $Nlog_2N$ کاهش می یابد.

این الگوریتمها را به روشهای متفاوتی می توان پیاده سازی کرد. دو روش ممکن آن پیاده سازی با تقسیم زمانی *decimation intime* و دیگری تقسیم فرکانسی *decimation in frequency* است که بصورت *in place* صورت می گیرد. در این روش حافظه خاص برای عملیات میانی مورد نیاز نیست.

### ۱-الف: الگوریتم مبنای ۲ تقسیم در میدان زمانی *Decimation in Time*

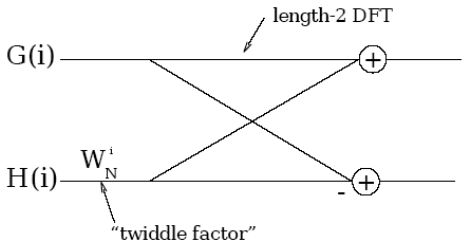
ابتدا به فلوگراف *2-DFT* توجه کنید که به نام پروانه *butterfly* خوانده می شود و ضریب آنـرا ضـریب چرخـشی *twiddle factor* مـی گویند.

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



EXAMPLE 2: Example DIT FFT Code

```
/*****************************/
/* fft: in-place radix-2 DIT
 DFT of a complex input
/* input: */
/* n: length of FFT: must
be a power of two */
/* m: n = 2**m */
/* input/output */
/* x: double array of length
n
with real part of  data
/* y: double array of length
 n with imag part of /*data */
/*****************************/
fft(n,m,x,y)
int n,m;
double x[],y[];
{
int i,j,k,n1,n2;
double c,s,e,a,t1,t2;
j = 0; /* bit-reverse */
n2 = n/2;
for (i=1; i < n - 1; i++)
{
n1 = n2;
while ( j >= n1 )
{
j = j - n1;
n1 = n1/2;
}
j = j + n1;
if (i < j)
{
t1 = x[i];
x[i] = x[j];
x[j] = t1;
t1 = y[i];
y[i] = y[j];
y[j] = t1;
}
}
n1 = 0; /* FFT */
n2 = 1;
for (i=0; i < m; i++)
{
n1 = n2;
n2 = n2 + n2;
e = -
6.283185307179586/n2;
a = 0.0;
for (j=0; j < n1; j++)
{
c = cos(a);
s = sin(a);
a = a + e;
for (k=j; k < n; k=k+n2)
{
t1 = c*x[k+n1] - s*y[k+n1];
t2 = s*x[k+n1] + c*y[k+n1];
x[k+n1] = x[k] - t1;
y[k+n1] = y[k] - t2;
x[k] = x[k] + t1;
y[k] = y[k] + t2;
}
}
}
return;
}
```




برای بالا بردن راندمان محاسباتی، $x(n)$ به ۲ دسته بـا $n$هـای زوج و فـرد تقـسیم گـشته و محاسـبات صورت می گیرد.

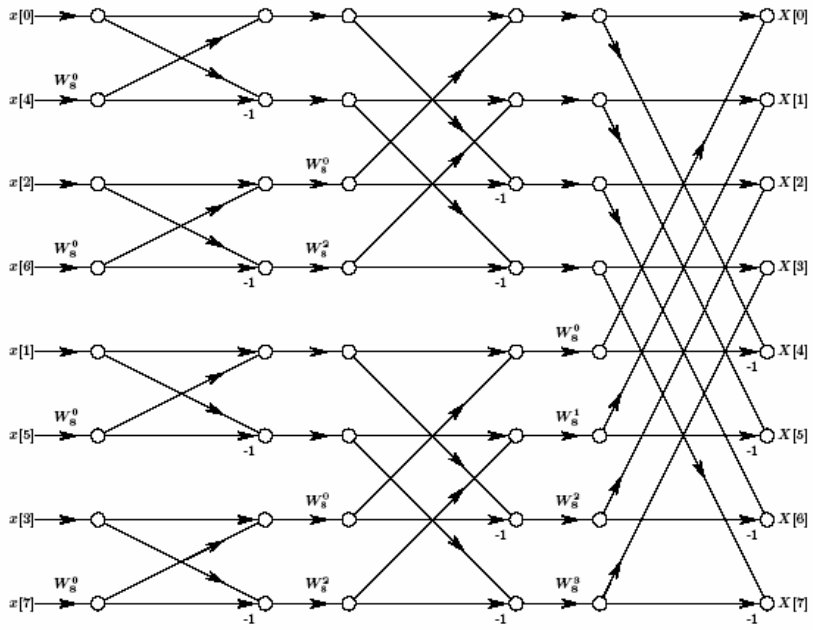
$$X[k] = \sum_{n=0}^{N-1} x[n]W_N^{kn} = \sum_{\substack{n=0 \\ n \text{ even}}}^{N-1} x[n]W_N^{kn} + \sum_{\substack{n=0 \\ n \text{ odd}}}^{N-1} x[n]W_N^{kn}$$

$$= \underbrace{\sum_{r=0}^{N/2-1} x[2r]W_{N/2}^{rk}}_{G[k]} + W_N^k \underbrace{\sum_{r=0}^{N/2-1} x[2r+1]W_{N/2}^{rk}}_{H[k]} \qquad W_N^2 = W_{N/2}$$

$$= G[k] + W_N^k H[k]$$

به این ترتیب

- تعداد ضرب $N$-DFT برابر $N^2$ است، که باتجزیه فـوق بـه ۲ عـدد $N/2$-DFT و $N$ ضـرب برابـر $2(N/2)^2+N<N^2$ میرسد. نامساوی برای $N>2$ صادق است
- $X(k)$ بایـــد بـــرای $k=0,\ldots,N-1$ محاســـبه گـــردد ولـــی بـــدلیل $G(k+N/2)=G(k)$ و $H(k+N/2)=H(k)$ تعداد محاسبات نصف می گردد.
- به همین ترتیب تعداد ضرایب چرخشی نیز بدلیل $W_N^{k+N/2}=-W_N^k$ به نصف کاهش می یابد.
- اگر این تجزیه در بلوکهای $N/2$-DFT تکرار شود تا بـه بلوکهـای $2$-DFT برسـد، در نهایـت هـر طبقه به بیش از $N/2$ ضرب نیاز ندارد. چون تعداد طبقات $\log_2 N$ است، تعـداد ضـربها بـه $N/2$ $\log_2 N$ کاهش می یابد.




با ملاحظه شکل نکته ای روشن می گردد که ورودیهای $x(n)$ بصورت مرتب اعمال نشده و لی خروجهای $X(k)$ مرتب دریافت شده اند به

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



این الگوریتمها تقسیم زمانی می گویند. ورودی یا خروجی الگوریتم که نا مرتب باشند ***bit-reversed*** می گویند. ضمنا الگوریتم ***In-Place*** است، یعنی طبقات میانی به حافظه اضلفی نیازندارند و خروجی هر طبقی در ورودی جای داده می شود.

روش دیگر نمایش عملیات فوق از اینقرار است. فرض کنید ***N=K=8*** باشد هدف بدست آوردن الگوریتم مبنای ۲ باشد تا

$$Y(k)=\sum_{n=0}^{N-1} x(n)W_N^{kn}$$

محاسبه گردد. حل ***n*** و ***k*** را بصورت ذیل می نویسیم و حاصلضرب ***kn*** را تعیین می کنیم.

$$k=2^2k_2+2^1k_1+2^0k_0 \qquad n=2^2n_2+2^1n_1+2^0n_0$$

$$kn=(4k_0)n_2+(4k_1+2k_0)n_1+(4k_2+2k_1+k_0)n_0$$

پریودیسیتی در طیف: ضربهائی که در آنها عددی بزرگتر از ***(N-1)*** در آنها ظاهر می گردند حذف می شوند چرا که برای مثال

$W_8^{16kn}=W_8^{0}=W_8^{8kn}$

به این ترتیب رابطه ریاضی برای طبقات اینگونه بدست می آید

$$x_1(k_0,n_1,n_0)=\sum_{n_2=0}^{1} x(n_2,n_1,n_0)\,W_8^{4k_0n_2}$$

طبقه اول : ۴ واحد

$$x_2(k_0,k_1,n_0)=\sum_{n_1=0}^{1} x_1(k_0,n_1,n_0)\,W_8^{(4k_1+2k_0)n_1}$$

طبقه دوم : ۴ واحد

$$x_3(k_0,k_1,k_2)=\sum_{n_0=0}^{1} x_2(k_0,k_1,n_0)\,W_8^{(4k_2+2k_1+k_0)n_0}$$

طبقه سوم : ۸ واحد

. جدول چگونگی کاهش محاسبات را نشان می دهد هرچه ***N*** افزایش یابد این کاهش حجم مقدار قابل ملاحظه ای به خود گرفته و در صد کاهش آن نیز افزایش می یابد.

| Number of Points $N$ | Complex Multiplications in Direct Computation $N^2$ | Complex Multiplications in FFT Algorithm $(N/2)\log_2 N$ | Speed Improvement Factor |
|---|---|---|---|
| 4 | 16 | 4 | 4.0 |
| 8 | 64 | 12 | 5.3 |
| 16 | 256 | 32 | 8.0 |
| 32 | 1,024 | 80 | 12.8 |
| 64 | 4,096 | 192 | 21.3 |
| 128 | 16,384 | 448 | 36.6 |
| 256 | 65,536 | 1,024 | 64.0 |
| 512 | 262,144 | 2,304 | 113.8 |
| 1,024 | 1,048,576 | 5,120 | 204.8 |

برنامه مقابل ***FFT*** را به روش تقسیم زمانی انجام می دهد..

**۱-ب: تقسیم در میدان فرکانس *Decimation in frequency***

این الگوریتم مشابه تقسیم زمانی است با این تفاوت که ورودی مرتب ولی خروجی نامرتب است.

برای این منظور منظور ابتدا ***DFT*** برای ***n***های زوج وسپس فرد محاسبه می گردد.

$$X[2r]=\sum_{n=0}^{N-1} x[n]W_N^{2rn}=\sum_{n=0}^{N/2-1} x[n]W_N^{2rn}+\sum_{n=N/2}^{N-1} x[n]W_N^{2rn}$$

$$=\sum_{n=0}^{N/2-1} x[n]W_N^{2rn}+\sum_{m=0}^{N/2-1} x[m+N/2]W_N^{2r(m+N/2)}$$

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

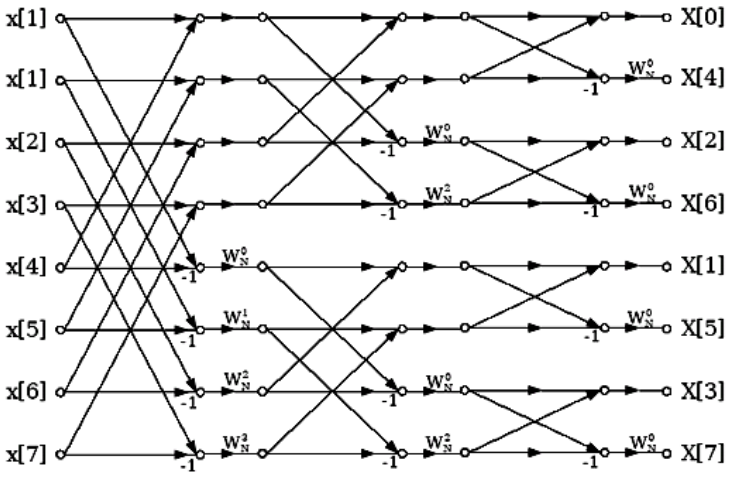


$$= \sum_{n=0}^{N/2-1} (x[n] + x[n+N/2])W_{N/2}^{rn} \qquad \text{using } W_N^{2rN/2} = 1 \text{ and } W_N^{2rn} = W_{N/2}^{rn}$$

برای فردها بدست می آید.

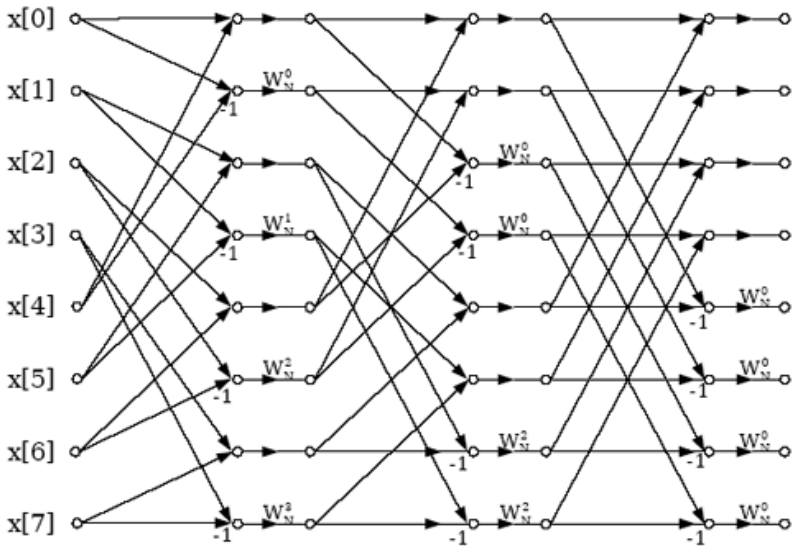
$$X[2r+1] = \sum_{n=0}^{N/2-1} W_N^n (x[n] - x[n+N/2])W_{N/2}^{rn}$$

شکل ذیل ساختار این الگوریتم را نشان می دهد.




## مبنای ۴

اگر $N=4^{\nu}$ باشد استفاده از الگوریتمهای مبنای ۴ مناسبتر است زیرا این الگوریتمها کاهش ۵۰ درصدی ضرب نسبت به همتای مبنا ۲ خود را در پی دارند در این الگوریتمها تعداد ضرب کمپلکس به $^{3}/_{8}N\log_2 N$ و تعداد جمع برابر $^{3N}/_{2}\log_2 N$ است. الگوریتمهای مبنای ۴ نیز ممکن است تقسیم در میدان زمانی یا تقسیم در میدان فرکانسی باشند.



www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



### دنباله های حقیقی

- دنباله *x(n)* و*y(n)* : اگر ۲ دنباله *x(n)* و *y(n)* در اختیار باشد بسیار مناسب است که *FFT* دنباله گرفته شود و سپس پاسخ فرکانسی هر یک بدین ترتیب استخراج گردد.

$$z(n) = x(n) + j\,y(n)$$
$$X_1(k) = \frac{1}{2}[z(k) + z^*(N-k)]$$
$$Y_1(k) = \frac{1}{2}[z(k) - z^*(N-k)]$$

- *DFT* دنباله حقیقی *g(n)* با طول *۲N*

برای این منظور دو دنباله به صورت $x_1(n)=g(n)$ و $x_2(n)=g(2n+1)$ تعریف می شوند. سپس *DFT* دنباله $x(n)=x_1(n)+jx_2(n)$ گرفته می شود. در نتیجه

$$X_1(k) = \frac{1}{2}[X(k) + X^*(N-k)]$$
$$X_2(k) = \frac{1}{2j}[X(k) - X^*(N-k)]$$

بدست می آید حال *DFT* دنباله *G(k)* برابر است با

$$G(k) = X_1(k) + W_{2N}^{k} X_2(k)$$
$$G(k+N) = X_1(k) - W_{2N}^{k} X_2(k) \qquad k = 0, 1, \dots, N-1$$

### محاسبه *IDFT*

برای محاسبه *IDFT* از همان الگوریتمهای *FFT* استفاده می شود با این تفاوت که $W_N^{kn}$ با $W_N^{-kn}$ تعویض و در نتیجه در *1/N* ضرب می گردد. محاسبه *IDFT* علاوه بر *N/2 log₂N* ضرب و *Nlog₂N* جمع کمپلکس به *N* ضرب و *N-1* جمع کمپلکس دیگر نیاز دارد

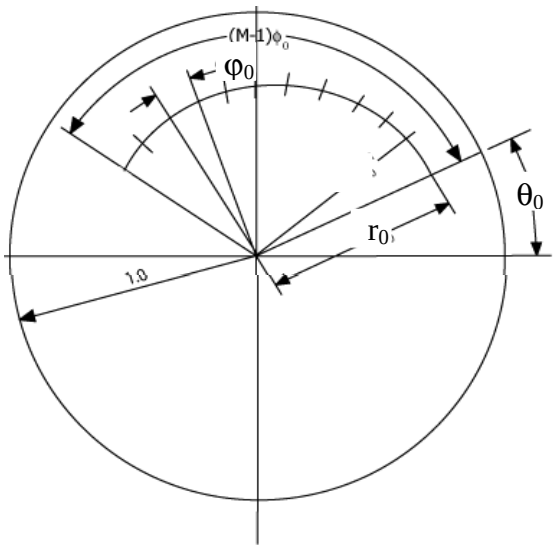
## ۸-۲-۳ تبدیل *Chirp Z*

فرض کنید هدف این باشد که *X(z)* را روی منحنی مشخصی محاسبه کنیم اگر این منحنی $z=e^{jw}$ باشد نتیجه همان *DFT* سیگنال *x(n)* است ولی اگر نقاط منحنی دیگری مثلا

$$z_k = r_0 e^{j\theta_0}\left(R_0 e^{j\varphi_0}\right)^k \qquad k = 0, 1, \dots, L-1$$

مدنظر باشد می توان نوشت.

$$X(z_k) = \sum_{n=0}^{N-1} x(n) z_k^{-n} = \sum_{n=0}^{N-1} x(n)(r_0 e^{j\theta_0})^{-n} V^{-nk}$$




بستگی به مقدار $r_0$ , $\Omega_0$ , $\theta_0$ , $\varphi_0$ بخشهائی از صفحه *z* که مشمول محاسبه می شوند مشخص می گردد. شکل این موضوع را نشان می دهد.

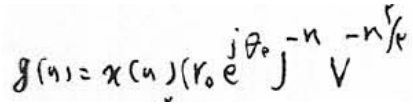
اگر $nk = \frac{1}{2}\left(n^2 + k^2 + (k-n)^2\right)$ و $V = R_0 e^{j\phi_0}$ قرار دهیم نتیجه می شود.

$$X(z_k) = \sum_{n=0}^{N-1} x(n)(r_0 e^{j\theta_0})^{-n} V^{-nk}$$
$$= V^{-k^2/2} \sum_{n=0}^{N-1} \left[x(n)(r_0 e^{j\theta_0})^{-n} V^{-n^2/2}\right] V^{(k-n)^2/2}$$

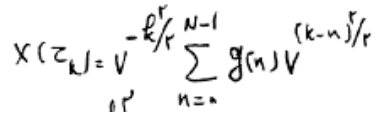
www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

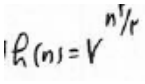


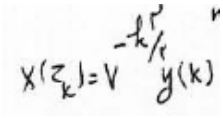
حال با تعریف



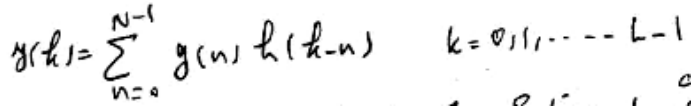
می نویسیم



این رابطه کانولوشن $g(n)$ با سیگنال  است که خروجی آن با $X(zk)$ رابطه



دارد که $y(k)$ برابر است با:

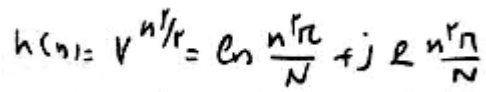


دلیل نام *chirp* برای این تبدیل آن است که اگر $R_0=1$ منظور شود فاز $h(n)$ برابر است $n\omega=n^2\varphi_0/2=n(n\varphi_0/2)$ مقدار $n\varphi_0/2$ فرکانس سیگنال کمپلکس اکسپونانسیل است که تابع زمان است و با آن افزایش می یابد. این سیگنال در رادار استفاده می شود و به آن سیگنال *chirp* می گویند.

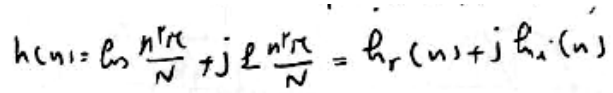
### محاسبه تبدیل *chirp Z*

محاسبه این تبدیل با *FFT* نیازمند $M\log_2 M$ ضرب کمپلکس است $M=N+L-1$ می باشد. $N$ تعداد نقاط سیگنال $x(n)$ و $L$ تعداد نقاطی است که در آن $x(zk)$ محاسبه می شود. اگر $L$ کوچک باشد محاسبه مستقیم با راندمان و اگر $L$ بزرگ باشد استفاده از *FFT* راندمان بالاتری دارد.

تبدیل *chirp Z* دارای پاسخ ضربه



است. وقتی $r_0=R_0=1$ و $\theta_0=0$ و $\varphi0=2\pi/N$ در نظر گرفته شود که در اینحال $X(zk)=X(k)$ است، *DFT* سیگنال $x(n)$ بدست می آید. این فیلتر را که *DFT* سیگنال $x(n)$ را محاسبه می کند بصورت سخت افزاری ساخته اند. این کار را دو فیلتر *FIR* با پاسخ ضربه $h_2(n)$ و $h_i(n)$ می سازند



### ۸-۲-۴ بهترین الگوریتم *FFT*

روشی که بیشتر مورد استفاده قرار می گیرد همان *FFT* در پایه ۲ است. *PFA* و *WFTA* به تعداد ضرب کمتری نیاز دارند ولی در کل در مقایسه با پیچیدگی اضافی آنها قابل ملا حظه نیست. جدول مقایسه ای بین روشهای مختلف ارا ئه می دهد.

| *Typical Numbers* | | | |
|---|---|---|---|
| | **N** | **M(real)** | **A(real)** |
| Radix 2 | 1024 | 10248 | 30728 |
| Split Radix | 1024 | 7172 | 27652 |
| Prime Factor algo | 1008 | 5804 | 29100 |
| Winograd FT algo | 1008 | 3548 | 34416 |

*WFTA* خصوصا بسیار مشکل است و عملا بندرت مورد استفاده قرار می گیرد. در کاربردهایی که انتخاب طول دنباله انتخابی است، طول به مقدار توانهای ۲ انتخاب می گردد. اما اگر این انتخاب محدودیت داشته باشد، *PFA* ترجیح داده می شود.

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



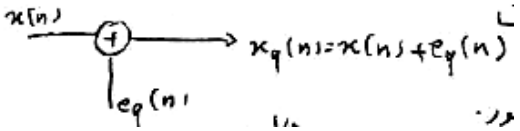
## ۸-۳ خطای ناشی از محدودیت رقمی

### ۸-۳-۱ خطای ناشی از کوانتیزه کردن سیگنال

در یک *A/D* ایده آل که خروجی *b* بیتی تولید می کند، خطای کوانتیزه کردن بین ورودی و خروجی ناشی از محدود بودن *b* وجود دارد. تاثیر این خطا در عملکرد سیستم را می توان با سیگنال نویز *e* که با ورودی جمع می شود در سیستم دخالت داد. اگر بزرگترین عددی که *A/D* نشان می دهد *R* باشد مقدار این خطا

$$e_q(n) = \pm \frac{1}{2}\frac{R}{2^{b+1}} = \pm\frac{\Delta}{2}$$

است که مدل ریاضی نحوه تاثیر آن بر سیستم مطابق شکل است.




در بررسی رفتار سیستم ناشی از این نویز معروضات ذیل در نظر گرفته می شود.


۱- توزیع $e_q(n)$ در محدوده مربوطه یکنواخت است.

۲- $e_q(n)$ از $e_q(m)$ که $m \neq n$ است مستقل هستند

۳- $e_q(n)$ از $x(n)$ مستقل هستند

### ۸-۳-۲ خطای ناشی از کوانتیزه شدن پارامترهای سیستم

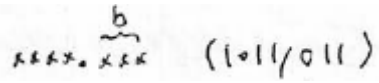
روشهای متداول اعداد مثبت و منفی از اینقرارند.

| *Number* | Sign-*Magnitude* | Offset *Binary* | Two's *Complement* | One's *Complement* |
|---|---|---|---|---|
| 7 | 0111 | 1111 | 0111 | 0111 |
| 6 | 0110 | 1110 | 0110 | 0110 |
| 5 | 0101 | 1101 | 0101 | 0101 |
| 4 | 0100 | 1100 | 0100 | 0100 |
| 3 | 0011 | 1011 | 0011 | 0011 |
| 2 | 0010 | 1010 | 0010 | 0010 |
| 1 | 0001 | 1001 | 0001 | 0001 |
| 0 | 0000 | 1000 | 0000 | 0000 |
| 0 | 1000 | 1000 | 0000 | 1111 |
| −1 | 1001 | 0111 | 1111 | 1110 |
| −2 | 1010 | 0110 | 1110 | 1101 |
| −3 | 1011 | 0101 | 1101 | 1100 |
| −4 | 1100 | 0100 | 1100 | 1011 |
| −5 | 1101 | 0011 | 1011 | 1010 |
| −6 | 1110 | 0010 | 1010 | 1001 |
| −7 | 1111 | 0001 | 1001 | 1000 |

پارامتر $a_k$ یک سیستم ممکن است توسط روش ممیز ثابت و یا ممیز شناور در عملیات ریاضی شرکت کند.

اعمال محدودیت رقمی ممکن است بصورت برشی *Truncation* و یا گرد کردن *Rounding* باشد.

**۱-الف) ممیز ثابت :**



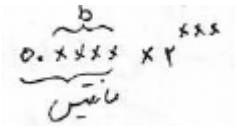

**خطای برشی** در این سبک به نحوه نمایش عدد منفی بستگی دارد. در روش *sign magnitude* این خطا برابر با $-2^{-b} \leq e_t \leq 2^{-b}$ و در *2'sconplement* برابر با $-2^{-b} \leq e_t \leq 0$ است.

**خطای گرد کردن** در ممیز ثابت برابر است با $2^{-b} \leq er \leq 2^{-b-1}$

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



**۱-ب) ممیز شناور** :

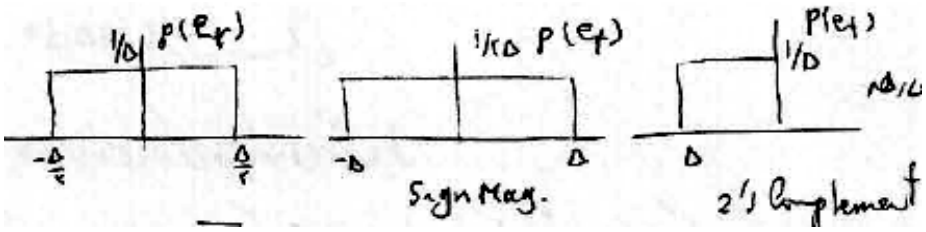


**خطای برشی** در این روش برای *2'scomplement* مانتیس $-2^{-b+1} \le e_r \le 0$، برای مقادیر مثبت و $0 \le er < 2^{-b+1}$ برای مقادیر منفی است.

**خطای کردن** مانیس برابر است با $-2^{-b} \le e_r \le 2^{-b}$

مانند حالت قبل این خطا را با نویزی با تابع یکنواخت در سیستم در نظر گرفته و با روش آماری رفتار سیستم در قبال آن بررسی می گردد.

شکل، توابع توزیع خطاها را نشان می دهد.




### ۸-۳-۳ بررسی حساسیت سیستمها به گرد کردن پارامترها

برای بررسی حساسیت به گردکردن، پارامتر $a_k$ را با $\bar{a}_k$ که $\bar{a}_k = ak + eak$ تعویض می نمایند. رفتار سیستم وقتی تغییر می کند که موقعیت صفرها یا قطبهای آن دستخوش تغییرات ناشی از گرد کردن در پارامترهای سیستم گردد. برای این منظور پارامتر انحراف قطب(صفر) اینگونه تعریف می شود.

$$\Delta P_i = \sum_{k=1}^{N} \frac{\partial P_i}{\partial \alpha_k} \cdot e_{\alpha k}$$

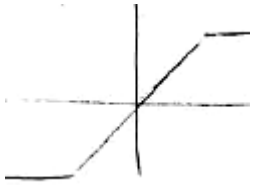
سیستمی که در آن تغییرات $\Delta P_i$ کمتر باشد سیستم مطمئن تری است. بررسی عملکرد سیستمها در مقابل خطای کوانتیزه پارامترها موارد ذیل را نتیجه داده است.

- نشان داده می شود که

$$\Delta P_i = -\sum_{k=1}^{N} \frac{P_i^{N-k}}{\prod_{\substack{l=1 \\ l \neq i}}^{N} (P_i - P_l)} \cdot \Delta a_k$$

است. برا ساس این رابطه، هرچه قطبهای سیستم به یکدیگر نزدیکتر باشد، حساسیت سیستم به گر شدن پارامترها بیشتر است.

- تجزیه سیستم به بلوک های تک قطبی و یا دو قطبی حساسیت را کاهش می دهد. اگر سیستم صفر نداشته باشد ترکیب موازی و یا سری جملات درجه ۱ یا ۲ روش های مناسبی هستند اگر سیستم دارای صفر باشد روش سری ترجیح دارد. خصوصا وقتی روش نمایش اعداد، ممیز ثابت است.
- در فیلترهای *FIR* که تمام صفر هستند اثر کوانتیزه تاثیری بر فاز سیستم نداشته و فقط دامنه را دستخوش تغییر می کند.
- روش نردبانی مشبک و یا مشبک به لحاظ اینکه پایداری سیستم را تامین می کند بسیار مناسب ولی تعداد محاسبات را افزایش می دهد.



- نوسان محدود *Limit Cycle* : (در سیستمهای *IIR*) : ناشی از گرد کردن حاصلضرب

در سیستم ممیز ثابت وقتی دو عدد درهم ضرب می شوند عدد بزرگتری بدست می آید که باید گرد گردد. این گرد کردن می تواند سیستم را به حالت نوسانی با دامنه محدود ببرد به نحوی که وقتی ورودی

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



ثابت است نوسان خروجی ادامه پیدا کند. این نوسان را باید با افزایش دقت رقمی از بین برد البته اعمال برش به جای گرد کردن می تواند آنرا متوقف کند ولی مشکلات دیگری ممکن است پدید آورد.

- نوسان محدود *(Limit Cycle)* : ناشی از سرریز *(overflow)* حاصلجمع در ممیز ثابت

وقتی چند عدد با هم جمع شوند عدد ممکن است آنقدر بزرگ شود که ظرفیت رقمی گنجایش آنرا نداشته باشد. در این حالت نیز نوسان با دامنه بزرگ در خروجی ایجاد می شود برای جلوگیری از آن می توان حاصلجمع را محدود کرد به عبارت دیگر جمع کننده اشباع شود. این عمل از ایجاد نوسان ناشی از سرریزی جلوگیری می کند ولی رفتار سیستم را غیرخطی و سیگنال را معوج می کند.

روش مناسبتر این است که دامنه سیگنال ورودی به نحو مطلوبی کاهش داده شود تا این مسئله اتفاق نیفتد به عبارت دیگر *dynamic Range* سیگنال در محدوده مناسبی قرار می گیرد. برای مثال اگر سیگنال با محدوده *±10* مشکل ایجاد کند سیگنال تضعیف شده تا محدوده آن به حد مناسبی مثلا *±2* یا هر محدوده مناسب دیگری کاهش داده شود.

### ۸-۳-۴ بررسی آماری اثرات گرد کردن ضرایب فیلتر

به دلیل زیاد بودن تعداد ضربها و جمعها و امکان بررسی دقیق اثر گرد کردنها وجود ندارد. لذا بررسی آماری مدنظر قرار می گیرد. برای مثال سیستم درجه ۱ را در نظر بگیرید.

$$y(n)=ay(n-1)+x(n)$$

بدلیل اثر گرد کردنها خروجی سیستم

$$v(n)=Q[a\ v(n-1)]+x(n)$$

می شود. اگر مدل گرد کردن

$$Q[a\ v(n-1)]=a\ v(n-1)+e(n)$$

در نظر گرفته شود

$$v(n)=av(n-1)+x(n)+e(n)=y(n)+q(n)$$

بدست می آید. اگر مفروضات آماری ذیل برقرار باشد

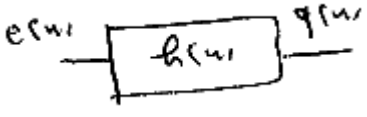
- تابع توزیع $e(n)$ یکنواخت با میانگین صفر و واریانس $\sigma_e^2=\frac{2^{-2b}}{12}$ باشد.
- $e(n)$ فرآیند سفید باشد $E[e(n)\ e(n+m)]=0 \quad if\ n\neq m$
- $E[e(n)\ x(n)]=0$ باشد

براساس ۳ می نویسیم .

پاسخ اصلی سیستم $y(n)=ay(n-1)+x(n)$

اثر نویز گرد کردن $q(n)=a\ y(n-1)+e(n)$

حال به سیستم نویز توجه کنید.




می دانیم که

$$\sigma_q^2=\sigma_e^2\cdot\frac{1}{2\pi}\int_{-\pi}^{\pi}|H(\omega)|^2 d\omega \quad , \quad m_q=m_e\,H(0)$$

این رابطه برای سیستم درجه ۱ با پاسخ ضربه $h(n)=a^n u(n)$ به ترتیب برابر با

$$m_q=m_e\times\frac{1}{1-a}$$

www.MatlabKar.com سایت متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب



$$\frac{\sigma_e^2}{1-a^2} = \sigma_q^2 = \sigma_e^2 \sum_{k=0}^{\infty} a^{2k}$$

می گردد.

$$\sigma_q^2 = \sigma_e^2 \sum_{k=-\infty}^{+\infty} h^2(k) = \sigma_e^2 \cdot \frac{1}{2\pi} \int_{-\pi}^{\pi} |H(\omega)|^2 d\omega \qquad \sigma_q^2 = \sigma_e^2 \sum_{k=0}^{\infty} a^{2k} = \frac{\sigma_e^2}{1-a^2}$$

### ۸-۳-۵ تاثیر محدودیت رقمی بر محاسبه *DFT*

محاسبه *DET* و هر عمل بردارش دیگری تحت تاثیر اثر کوانیزه سیگنال ورودی ناشی از *A/D* محدودیت رقمی پارامترها و اعمال این محدودیت بر حاصلجمع و حاصلضرب اعداد در یکدیگر قرار دارد که در نهایت خروجی را تحت تاثیر قرار می دهد. در تمامی موارد عوامل خطا بصورت نویزی با توزیع یکنواخت در نظر گرفته می شود و از نظر آماری تاثیر آن برعملکرد کلی عمل پردازش مورد بررسی قرار می گیرد.