دانشگاه تهران

دانشکده‌ی فنی

گروه مهندسی برق و کامپیوتر

عنوان:

شبیه‌سازی و مقایسه روش‌های همزمان‌سازی سمبل و دنبال کردن آنها برای مدولاسیون‌های خطی

نگارش:

مرتضی بناگر

استاد راهنما:

دکتر علی الفت

استاد داور:

دکتر مریم صباغیان

پایان‌نامه برای دریافت درجه‌ی کارشناسی در رشته‌ی

مهندسی برق-گرایش مخابرات

شهریور ماه ۱۳۹۱

چکیده

یکی از مهمترین عناصر یک سیستم مخابراتی قسمت همزمان‌سازی آن است که شامل دو بخش اساسی بازیابی زمان و بازسازی حامل می‌باشد. در این پروژه پس از معرفی تئوری‌های موبوط به تخمین پارامتر و فیلتر منطبق، روی روش‌های مختلف بازیابی زمان بحث می‌کنیم و با انجام شبیه‌سازی‌های لازم، سعی در جبران آفست زمانی داریم.

جدول اختصارات

MPSK	M-ary Phase Shift Keying
PAM	Pulse Amplitude Modulation
ML	Maximum Likelihood
MAP	Maximum A Posteriori Probability
MMSE	Minimum Mean-Square-Error
MSE	Mean-Square-Error
VCO	Voltage Controlled Oscillator
ELG	Early-Late Gate
RRC	Root Raised-Cosine
RC	Raised-Cosine
SER	Symbol Error Rate

فهرست شکل‌ها

عنوان .... صفحه

فهرست مطالب

۱- مقدمه

در مخابرات دیجیتال اطلاعاتی که فرستاده می‌شوند، با تأخیری که ممکن است نامشخص باشد، در گیرنده دریافت می‌شوند. اطلاعات دریافتی باید پس از دمدولاتور[۱]، به طور متناوب (با نرخ سمبل) نمونه‌برداری شوند تا اطلاعات ارسالی حاصل شوند. حال اگر زمان یا فرکانس نمونه‌برداری به درستی انتخاب نشوند، اطلاعاتی که ما تشخیص می‌دهیم نیز اشتباه خواهند بود. بنابراین در اینجا دو مسأله مطرح است: همزمان‌سازی سمبل[۲] و همزمان‌سازی حامل[۳]. همزمان‌سازی سمبل در تمامی سیستم‌های مخابراتی که داده‌ها را به صورت سنکرون ارسال می‌کنند، نیاز است؛ در حالی که بازسازی حامل تنها زمانی لازم است که می‌خواهیم سیگنال را به صورت همدوس[۴] آشکار کنیم [۱].

مسأله‌های همزمان‌سازی دیگری نیز وجود دارد که عبارتند از همزمان‌سازی کلمه[۵]، قالب[۶] و بسته[۷]. عاملی که موارد اخیر را از همزمان‌سازی سمبل و حامل متمایز می‌کند، قابل حل بودن آنها به کمک خود پیغام است. در حقیقت می‌توان تعدادی از بیت‌ها را در فرستنده به فرم خاصی تکرار کرد تا دیگر نگران مسأله‌ی همزمان‌سازی در گیرنده نباشیم [۲]. هدف ما در این پروژه بررسی همزمان‌سازی سمبل خواهد بود.

به طور کلی، در گیرنده‌های مخابرات دیجیتال چهار مرحله‌ی اساسی داریم:

۱) کنترل بهره‌ی خودکار (AGC[۸]): توان سیگنال را به مقدار مشخصی می‌رساند و غالباً در حوزه‌ی آنالوگ بررسی می شود.

۲) بازسازی زمان یا ساعت[۹] (همزمان‌سازی سمبل): تعیین دو کمیت فرکانس و فاز نمونه‌برداری.

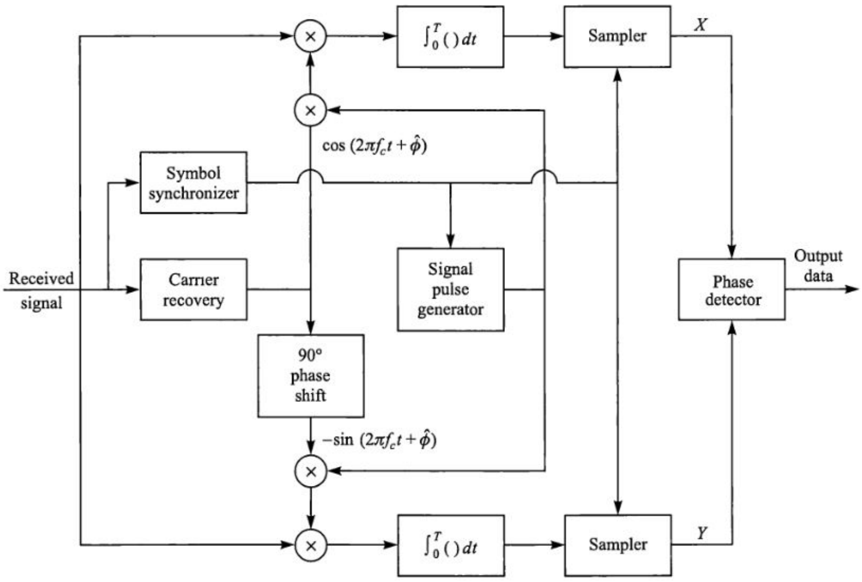
۳) بازسازی حامل[۱۰] (همزمان‌سازی حامل): از بین بردن آفست فرکانس که در مرحله‌ی دمدولاسیون به منظور پردازش درست سیگنال در باند پایه ایجاد می‌شود.

۴) برابرسازی کانال[۱۱]: حذف تداخل‌های بین سمبل (ISI[۱۲]) به کمک فیلترهای وفقی.

[1] Demodulator
[2] Symbol Synchronization
[3] Carrier Synchronization
[4] Coherent
[5] Word Synchronization
[6] Frame Synchronization
[7] Packet Synchronization
[8] Automatic Gain Control
[9] Timing or Clock Recovery
[10] Carrier Recovery
[11] Channel Equalization
[12] Inter-Symbol Interference

در این پروژه ما مرحله‌ی دوم را مورد بررسی قرار می‌دهیم و معیار بیشینه شباهت (ML[۱۳]) را به عنوان نقطه‌ی شروع ارزیابی می‌کنیم. سپس با تحلیل تئوری الگوریتم های موجود و شبیه‌سازی آنها، نتایج لازم را ارائه خواهیم داد.

در زیر شمایی از نمودار بلوکی یک گیرنده‌ی مدولاسیون MPSK[۱۴] شامل بازسازی حامل و همزمان‌سازی سمبل آمده است [۱].



شکل ۱ – گیرنده‌ی مدولاسیون MPSK شامل بازسازی حامل و ساعت

همانطور که در شکل نشان داده شده است، همسان‌ساز سمبل، نمونه‌بردار و خروجی مولد سیگنال را کنترل می‌کند. بلوک بازسازی حامل نیز با تخمین فاز $\hat{\phi}$، ورودی دو مقایسه‌کننده[۱۵] (یا فیلتر منطبق[۱۶]) را مهیا می-سازد.

[13] Maximum Likelihood
[14] M-ary Phase Shift Keying
[15] Correlator
[16] Matched Filter

۲- مدل ریاضی و تخمین پارامتر[17]

فرض کنید فرستنده سیگنال $s(t)$ را ارسال می‌کند و گیرنده آنرا با تأخیر کانال و نویز به صورت زیر دریافت می‌کند:

$$r(t) = s(t-\tau) + n(t)$$

می‌توان از معادل پایین‌گذر سیگنال‌های میان‌گذر بهره برد و رابطه‌ی بالا را این‌گونه نوشت:

$$r(t) = Re\{[s_l(t-\tau)e^{j\phi} + z(t)]e^{j2\pi f_c t}\}$$

که در این رابطه τ تأخیر کانال، $\phi = -2\pi f_c \tau$ فاز حامل و $z(t)$ و $s_l(t)$ به ترتیب معادل‌های پایین‌گذر سیگنال‌های $n(t)$ و $s(t)$ می‌باشند. رابطه‌ی یک سیگنال میان‌گذر با معادل پایین‌گذرش به این صورت است:

$$s(t) = Re\{s_l(t)e^{j2\pi f_c t}\} \qquad or \qquad s_l(t) = (s(t) + j\hat{s}(t))e^{-j2\pi f_c t}$$

در این رابطه $\hat{s}(t)$ نمایانگر تبدیل هیلبرت $s(t)$ می‌باشد.

در عمل، فاز حامل تنها به تأخیر کانال بستگی ندارد و عوامل متعددی همچون همزمان نبودن نوسان-ساز محلی در گیرنده با نوسان‌ساز فرستنده باعث می‌شوند که برای محاسبه‌ی $r(t)$ باید هر دو متغیر τ و ϕ تخمین زده شوند [۱].

حال می‌توان سیگنال دریافتی را این‌گونه نوشت:

$$r(t) = s(t;\tau,\phi) + n(t)$$

به طور کلی دو معیار اساسی به منطور تخمین پارامتر سیگنال وجود دارد: معیار بیشینه شباهت و معیار بیشینه احتمال پسین[18]. در اولی بردار پارامتر سیگنال $\boldsymbol{\theta}$ به صورت قطعی (غیر تصادفی) اما نامشخص مدل می‌شود در حالی که در دومی $\boldsymbol{\theta}$ به صورت تصادفی و با یک چگالی احتمال پیشین[19] $f(\boldsymbol{\theta})$ مدل خواهد شد.

فرض کنید سیگنال $r(t)$ را به کمک N تابع عمود بر هم و با انرژی واحد $\{\phi_n(t)\}$ بسط داده‌ایم و به بردار ضرایب $\boldsymbol{r} = (r_1, r_2, \dots, r_N)$ رسیدیم. چگالی احتمال مشترک این ضرایب را $f(\boldsymbol{r}|\boldsymbol{\theta})$ در نظر بگیرید. میزانی از $\boldsymbol{\theta}$ که این تابع را بیشینه کند، تخمینگر ML نامیده می‌شود. از طرف دیگر، میزانی از $\boldsymbol{\theta}$ که $f(\boldsymbol{\theta}|\boldsymbol{r})$ را که با رابطه‌ی زیر داده می‌شود، بیشینه کند، تخمینگر MAP نامیده می-شود.

---

[17] Parameter Estimation
[18] Maximum A Posteriori Probability (MAP)
[19] A Priori Probability

$$f(\boldsymbol{\theta}|\boldsymbol{r}) = \frac{f(\boldsymbol{r}|\boldsymbol{\theta})f(\boldsymbol{\theta})}{f(\boldsymbol{r})}$$

حال اگر هیچ اطلاعی از $\boldsymbol{\theta}$ نداشته باشیم، می‌توانیم فرض کنیم که یکنواخت است و تأثیری در محاسبه‌ی بالا ندارد و در نتیجه، در این حالت این دو تخمینگر عملکرد یکسانی خواهند داشت. ثابت می‌شود که تخمینگر MAP بهینه است، اما ما در اینجا به دو دلیل از آن استفاده نمی‌کنیم: اول اینکه پیچیدگی آن از ML بیشتر است، حال آنکه مزیت آن چندان چشمگیر نیست! نکته‌ی دوم در این است که ما پارامترهای τ, ϕ را به صورت غیر تصادفی و نامشخص در نظر می‌گیریم که خود تخمینگر ML را پیشنهاد می‌کند [۳].

با توجه به توصیف بالا و اینکه نویز جمع شونده را سفید و گوسی با میانگین صفر و چگالی طیف توان N_0 در نظر می‌گیریم، می‌توانیم تابع چگالی احتمال $f(\boldsymbol{r}|\boldsymbol{\theta})$ را به صورت زیر بنویسیم:

$$f(\boldsymbol{r}|\boldsymbol{\theta}) = \left(\frac{1}{\sqrt{\pi N_0}}\right)^N \exp\left\{-\sum_{n=1}^{N} \frac{[r_n - s_n(\boldsymbol{\theta})]^2}{N_0}\right\}$$

که در این رابطه داریم (T_0 در اینجا بازه‌ی انتگرال‌گیری برای بسط $r(t)$ و $s(t;\boldsymbol{\theta})$ می‌باشد):

$$r_n = \int_{T_0} r(t)\phi_n(t)dt$$

$$s_n(\boldsymbol{\theta}) = \int_{T_0} s(t;\boldsymbol{\theta})\phi_n(t)dt$$

حال ثابت می‌کنیم که عبارت داخل تابع نمایی را می‌توان (به صورت حدی) بر حسب خود توابع $r(t)$ و $s(t;\boldsymbol{\theta})$ به شکل زیر نوشت:

$$\lim_{N\to\infty} \sum_{n=1}^{N} \frac{[r_n - s_n(\boldsymbol{\theta})]^2}{N_0} = \frac{1}{N_0}\int_{T_0} [r(t) - s(t;\boldsymbol{\theta})]^2\, dt$$

در واقع با توجه به بسط توابع $r(t)$ و $s(t;\boldsymbol{\theta})$ داریم:

$$r(t) = \sum_{n=1}^{N} r_n\, \phi_n(t) \qquad s(t;\boldsymbol{\theta}) = \sum_{n=1}^{N} s_n(\boldsymbol{\theta})\, \phi_n(t)$$

$$\frac{1}{N_0}\int_{T_0}[r(t)-s(t;\boldsymbol{\theta})]^2\,dt$$

$$=\frac{1}{N_0}\int_{T_0}[\sum_{n=1}^{N}(r_n-s_n(\boldsymbol{\theta}))\,\phi_n(t)]^2\,dt$$

$$=\frac{1}{N_0}\int_{T_0}\sum_{n=1}^{N}\sum_{m=1}^{M}(r_n-s_n(\boldsymbol{\theta}))\,(r_m-s_m(\boldsymbol{\theta}))\,\phi_n(t)\,\phi_m(t)dt$$

و از آنجایی که توابع $\phi_n(t)$ بر هم عمود و با انرژی واحد هستند، انتگرال حاصلضرب دو تایی آنها صفر خواهد بود، مگر اینکه اندیس‌هایشان یکسان باشد. پس خواهیم داشت:

$$=\frac{1}{N_0}\sum_{n=1}^{N}\sum_{m=1}^{M}(r_n-s_n(\boldsymbol{\theta}))\,(r_m-s_m(\boldsymbol{\theta}))\,\delta_{nm}$$

$$=\frac{1}{N_0}\sum_{n=1}^{N}[r_n-s_n(\boldsymbol{\theta})]^2$$

که همان نتیجه‌ی مطلوب است.

حال با صرف نظر از ضریب ثابت در $f(\boldsymbol{r}|\boldsymbol{\theta})$، تابع شباهت[۲۰] را تشکیل می‌دهیم:

$$\Lambda(\boldsymbol{\theta})=\exp\{-\frac{1}{N_0}\int_{T_0}[r(t)-s(t;\boldsymbol{\theta})]^2\,dt\}$$

واضح است که کمینه کردن این انتگرال، معادل بیشینه کردن $f(\boldsymbol{r}|\boldsymbol{\theta})$ است.

این انتگرال فاصله‌ی فضای سیگنال بین دو تابع $r(t)$ و $s(t;\boldsymbol{\theta})$ که در بازه‌ی T_0 تعریف شده‌اند را نشان می‌دهد [۲]، [۳].

با ساده‌سازی $L(\boldsymbol{\theta})$ داریم:

$$L(\boldsymbol{\theta})=\exp\left\{-\frac{1}{N_0}\int_{T_0}r^2(t)\,dt-\frac{1}{N_0}\int_{T_0}s^2(t;\boldsymbol{\theta})\,dt+\frac{2}{N_0}\int_{T_0}r(t)s(t;\boldsymbol{\theta})\,dt\right\}$$

جمله‌ی اول عبارت بالا مستقل از $\boldsymbol{\theta}$ می‌باشد و جمله‌ی دوم نیز حاکی از انرژی سیگنال است که به $\boldsymbol{\theta}$ وابسته نیست (برای T_0 های بزرگ، این انتگرال تغییرات بسیار ناچیزی با $\boldsymbol{\theta}$ خواهد داشت که از آن صرف نظر می‌کنیم [۲]). پس می‌توان آنها را ثابت در نظر گرفت و تابع شباهت را این‌گونه نوشت:

---

[20] Likelihood Function

$$L(\boldsymbol{\theta}) = K \exp\left\{\frac{2}{N_0}\int_{T_0} r(t)s(t;\boldsymbol{\theta})\,dt\right\}$$

بدون کاسته شدن از کلیت مسأله، K را یک در نظر می‌گیریم و از عبارت بالا لگاریتم می‌گیریم تا به تابع لگاریتم شباهت[۲۱] برسیم.

$$\Lambda(\boldsymbol{\theta}) = \ln L(\boldsymbol{\theta}) = \frac{2}{N_0}\int_{T_0} r(t)s(t;\boldsymbol{\theta})\,dt$$

این انتگرال همبستگی[۲۲] بین سیگنال دریافتی $r(t)$ و سیگنال مرجع $s(t;\boldsymbol{\theta})$ را نشان می‌دهد و تخمین ML برای $\boldsymbol{\theta}$، در واقع مقداری از $\boldsymbol{\theta}$ است که تابع $\Lambda(\boldsymbol{\theta})$ را بیشینه کند.

هنگامی که $s(t;\boldsymbol{\theta})$ حاوی پیغام تصادفی باشد، تابع شباهت مناسب برای تخمین $\boldsymbol{\theta}$ با میانگین‌گیری از $L(\boldsymbol{\theta})$ (و نه $\Lambda(\boldsymbol{\theta})$) حاصل خواهد شد [۲].

در ادامه ابتدا نگاهی به تئوری فیلتر منطبق خواهیم داشت و سپس با کمک معیارهای مطرح شده به بررسی گیرنده‌های شامل همسان‌ساز سمبل می‌پردازیم.

[21] Log-Likelihood Function
[22] Correlation

۳-عملکرد فیلتر منطبق در گیرنده

سیستم‌های مخابراتی باید نسبت به نویز و دیگر تداخل‌هایی که در کانال موجود است مقاوم باشند. هدف فیلتر منطبق نیز کاهش حساسیت سیگنال نسبت به نویز می‌باشد [۴].

حال فرض کنید که سیگنال دریافتی $r(t)$ با نویز سفید گوسی $n(t)$ با چگالی طیف توان η جمع می‌شود و سیگنال $s(t)$ حاصل می‌شود.

$$s(t) = r(t) + n(t)$$

این سیگنال را از فیلتر دیگری (با پاسخ ضربه $h_r(t)$) می‌گذرانیم و هدف ما در تعیین این فیلتر این است که نسبت توان سیگنال به توان نویز در خروجی این گیرنده بیشینه شود؛ در واقع تا حد ممکن نویز محدود شود. خروجی فیلتر را این‌گونه نام‌گذاری می‌کنیم:

$$y(t) = y_r(t) + y_n(t)$$

که $y_r(t)$ و $y_n(t)$ به ترتیب حاصل عبور $r(t)$ و $n(t)$ از فیلتر می‌باشند.

توان نویز پس از عبور از فیلتر چنین خواهد شد[۲۳]:

$$P_n = \int_{-\infty}^{\infty} \mathcal{P}_n(f)df = \int_{-\infty}^{\infty} \eta |H_r(f)|^2 df$$

از طرفی برای سیگنال $r(t)$ نیز داریم:

$$y_r(\tau) = \int_{-\infty}^{\infty} Y_r(f)e^{j2\pi f\tau}dt = \int_{-\infty}^{\infty} R(f)H_r(f)e^{j2\pi f\tau}dt$$

بنابراین معیاری که باید بیشینه شود، این‌گونه خواهد بود:

$$\frac{|y_r(\tau)|^2}{P_n} = \frac{\left|\int_{-\infty}^{\infty} R(f)H_r(f)e^{j2\pi f\tau}dt\right|^2}{\int_{-\infty}^{\infty} \eta |H_r(f)|^2 df}$$

که اعمال نامساوی کوشی-شوارتز[۲۴]، حد بالایی برای عبارت بالا به این صورت بدست می‌دهد:

$$\frac{|y_r(\tau)|^2}{P_n} \leq \frac{\left(\int_{-\infty}^{\infty} |H_r(f)|^2 df\right)\left(\int_{-\infty}^{\infty} \left|R(f)e^{j2\pi f\tau}\right|^2 dt\right)}{\int_{-\infty}^{\infty} \eta |H_r(f)|^2 df} = \frac{1}{\eta}\int_{-\infty}^{\infty} |R(f)|^2 dt$$

که تساوی در شرایط زیر حاصل می‌شود:

$$H_r(f) = k\left(R(f)e^{j2\pi f\tau}\right)^*$$

[۲۳] در تمام متن، از حروف کوچک برای نمایش حوزه‌ی زمان سیگنال و از حروف بزرگ برای نمایش تبدیل فوریه‌ی آن استفاده شده است؛ مگر اینکه خلاف آن ذکر شود.

[24] Cauchy–Schwarz Inequality

برای توضیح بیشتر و اثبات به پیوست A در [۴] مراجعه کنید.

و در حوزه‌ی زمان:

$$\mathcal{F}^{-1}\big(H_r(f)\big) = h_r(t) = k.r^*(\tau - t)$$

که اگر $r(t)$ حقیقی هم باشد، علامت مزدوج را می‌توان برداشت.

همانطور که مشاهده می‌شود، خروجی فیلتری با پاسخ ضربه بالا، SNR را در زمان‌های $t = \tau$ بیشینه می‌کند و از آنجا که این پاسخ ضربه، در واقع همان معکوس شده‌ی $r(t)$ در زمان است، می‌گویند که این فیلتر بر سیگنال ورودی *منطبق* است و به همین دلیل نامش را "فیلتر منطبق" گذاشته‌اند.

برای روشن‌تر شدن موضوع، شبیه‌سازی زیر را انجام دادیم که در آن حالت‌هایی که فیلتر گیرنده با داده‌ها منطبق باشد و یا نباشد، مورد ارزیابی قرار گرفته‌اند.

```
% Matched Filter Performance in the Receiver %

% Setting Variables %

fs = 24e6;
fd = 1e6;
N = 1000;
%t = -14.4:0.2:14.4;

% Designing filters %

g = rcosine(fd,fs,'fir/normal');   % Raised cosine as our pulse shaping filter
gnorm = sqrt(sum(g.^2));
g = g/gnorm;                       % Normalize it

% Recieved filter is matched to the transmitted one
r = rcosine(fd,fs,'fir/normal');
%r = sinc(t);                      % Recieved filter is not matched (case 1)
%r = rcosine(fd,fs,'fir/sqrt');    % Recieved filter is not matched (case 2)
rnorm = sqrt(sum(r.^2));
r = r/gnorm;

% Data and Noise %

m = sign(randn(N));
Lm = length(m);
n = 0.1*randn(Lm);
f = filter(g,1,m);                 % Passing the data through pulse shaping filter
DataMatch = filter(fliplr(r),1,f); % Passing it through the matched filter
```

$NoiseMatch$
$= filter(fliplr(r),1,n);$ % Passing noise through the matched filter
$PowData = sum(DataMatch.^2)$
$/length(DataMatch);$ % Computing the power of signal
$PowNoise = sum(NoiseMatch.^2)$
$/length(NoiseMatch);$ % Computing the power of noise
$SNR = PowData/PowNoise;$ % Signal to noise ratio

خروجی این کد در حالات مختلف در جدول زیر آمده است:

Filter	***Output SNR***
$rcosine(fd,fs,'fir/normal');$ ***Which is matched***	**2084**
$rcosine(fd,fs,'fir/sqrt');$ ***Which is not matched***	**1974**
$sinc(t)$ ***Which is not matched***	**464**

همانطور که روشن است، SNR زمانی بیشینه است که فیلتر گیرنده، فیلتر منطبق باشد و این مسأله کاملاً مستقل از این است که نویز و یا فیلتر شکل‌دهنده‌ی اولیه چه باشد.

در ادامه به بررسی دقیق‌تر مسأله‌ی همزمانی سمبل می‌پردازیم.

۴- مسأله‌ی همزمان‌سازی سمبل

هنگامی که سیگنال به گیرنده می‌رسد، به فرم یک سیگنال آنالوگ پیچیده می‌باشد که باید از آن در زمان‌های مناسب و به طور متناوب با نرخ سمبل[۲۵] نمونه‌برداری کنیم تا پیام ارسالی بازسازی شود. برای انجام چنین عملی، نیاز به یک سیگنال ساعت در گیرنده داریم که به استخراج این سیگنال در گیرنده بازسازی زمان یا همزمان‌سازی سمبل می‌گوییم. بهترین زمان برای نمونه‌برداری، زمانی در بازه‌ی یک سمبل است که خروجی فیلتر گیرنده در آنجا بیشینه باشد.

روش‌های متعددی برای همزمان‌سازی سمبل وجود دارد که به آنها اشاره می‌کنیم:

۱) وجود یک ساعت مرجع: فرستنده و گیرنده با یک ساعت مرجع که یک سیگنال زمانی بسیار دقیق تولید می‌کند، همزمان می‌شوند. این روش در فرکانس‌های بسیار پایین[۲۶] (زیر 30^{kHz}) کاربرد دارد و پرهزینه است.

۲) ارسال فرکانس و زمان نمونه‌برداری به همراه سیگنال پیام: گیرنده می‌تواند به کمک یک فیلتر باند-باریک که روی فرکانس ساعت گیرنده تنظیم شده است، سیگنال ساعت را استخراج کند. مزیت عمده‌ی این روش سادگی آن است.

اشکالات این روش نیز این است که فرستنده باید مقداری از توان ارسالی و پهنای باند موجود خود را صرف ارسال این سیگنال ساعت کند. با این حال این روش در حالاتی که تعداد کاربران زیاد است، کاربرد دارد. چرا که مصرف این توان و پهنای باند به نسبت تعداد کاربران کاهش می‌یابد.

۳) همزمان‌سازی به کمک خود سیگنال دریافتی (خود-همزمان‌سازی[۲۷]): روش‌های مختلفی برای این کار وجود دارد که آنها را بررسی خواهیم کرد.

در ادامه ابتدا روش‌های مختلف خود-همزمان‌سازی سمبل را – که به دو دسته‌ی کلی non-decision-directed و decision-directed تقسیم می‌شوند – شرح داده و در هر کدام پس از معرفی نمودارهای بلوکی مربوطه، به بررسی الگوریتم‌هایی برای یافتن آفست زمانی (τ) می‌پردازیم.

از آنجایی که محور اصلی پروژه روی روش‌های non-decision-directed می‌گردد، شبیه‌سازی‌های ما نیز در همان محدوده خواهد بود.

---

[25] Symbol Rate
[26] Very Low Frequency (VLF)
[27] Self-Synchronization

۴-۱- بازسازی زمان ***decision-directed***

در این حالت، خروجی دمدولاتور را به عنوان اطلاعات ارسالی درست در نظر می‌گیریم. در واقع فرض می‌کنیم، که می‌دانیم چه ارسال شده (احتمال خطای صفر) و در پی یافتن آفست زمانی مورد نظر هستیم. از آنجایی که اصولاً در مخابرات، گیرنده از داده‌های ارسالی اطلاعی ندارد، این روش چندان عملی به نظر نمی‌رسد. با این وجود، با اعمال تغییرات زیر می‌توان آن را مورد استفاده قرار داد:

۱) ارسال داده‌های آموزشی[۲۸] (اطلاعات قراردادی ای که هم گیرنده و هم فرستنده می‌دانند) قبل از ارسال داده‌های اصلی

۲) بالا بودن نسبت سیگنال به نویز E_b/N_0

اگر داده‌های آموزشی به اندازه‌ی کافی زیاد باشند، گیرنده می‌تواند روی آفست زمانی قفل کند. از طرف دیگر اگر E_b/N_0 بزرگ باشد، احتمال خطا کوچک خواهد بود و در نتیجه پس از داده‌های آموزشی نیز، گیرنده می‌تواند به خوبی آفست زمانی را دنبال کند.

نمادگذاری زیر را برای ادامه‌ی بحث، در نظر بگیرید:

اطلاعات $a[n]$ یا a_n با عبور از فیلتر فرستنده[۲۹] $g_T(t)$، وارد کانال $c(t)$ می‌شوند و در گیرنده پس از گذشتن از فیلتر گیرنده (فیلتر منطبق) $g_R(t)$، سیگنال $y(t)$ را تشکیل می‌دهد. با نمونه‌برداری صحیح از این سیگنال، داده‌های اولیه بازیابی می‌شوند. برای سادگی، ترکیب فیلترهای فرستنده و گیرنده و کانال را $x(t)$ می‌نامیم؛ یعنی:

$$x(t) = g_T(t) * c(t) * g_R(t)$$

نویز را نیز در خروجی فیلتر منطبق $n(t)$ در نظر می‌گیریم.

حال فرض کنید که سیگنال باند پایه در گیرنده به فرم زیر باشد:

$$v(t) = \sum_{n=-\infty}^{\infty} a_n g_T(t - nT)$$

که در آن T بازه‌ی یک سمبل است. سیگنال $y(t)$ نیز چنین خواهد بود:

$$y(t) = \sum_{n=-\infty}^{\infty} a_n x(t - nT - \tau_0) + n(t)$$

[28] Training Sequence

[۲۹] واضح است که منظور، پاسخ ضربه‌ی فیلتر فرستنده است.

حال به کمک این نمادگذاری، به بررسی هر یک از روش‌ها می‌پردازیم.

۴-۱-۱- روش کمینه میانگین مربع خطا ($MMSE$[۳۰])

این روش با کمینه کردن میانگین مربع خطا بین خروجی فیلتر گیرنده و سمبل‌های مطلوب، سعی در تخمین آفست زمانی τ_0 دارد.

خروجی فیلتر گیرنده در بازه‌ی mام – که وابسته به τ است – را می‌توانیم این‌گونه بنویسیم:

$$y_m(\tau_0) = \sum_{n=-\infty}^{\infty} a_m x(mT - nT - \tau_0) + n(mT)$$

حال از خروجی آشکارساز اطلاعات را بازیابی کنیم ($\hat{a}_m$) و به کمک این اطلاعات، داریم:

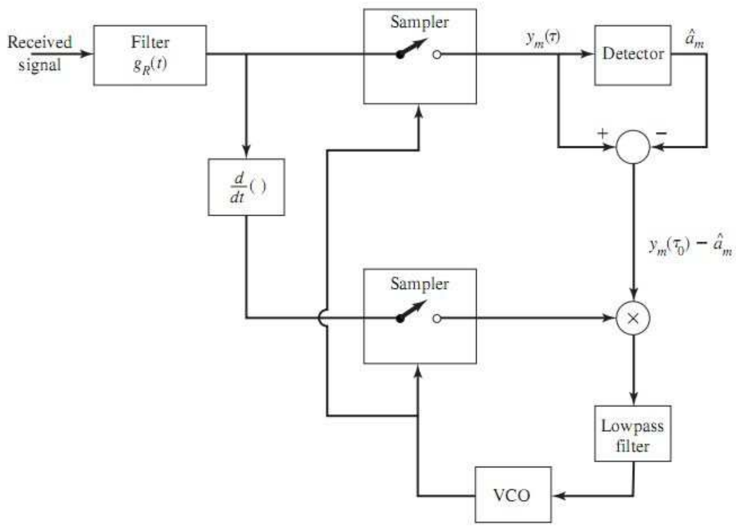
$$\mathrm{MSE} = E\{[y_m(\tau_0) - \hat{a}_m]^2\}$$

با صفر قرار دادن مشتق این عبارت نسبت به τ_0 خواهیم داشت:

$$\sum_m [y_m(\tau_0) - \hat{a}_m] \frac{dy_m(\tau_0)}{d\tau_0} = 0$$

که بیانگر این نکته است که بهترین زمان نمونه‌برداری در شرایطی حاصل می‌شود که سیگنال خطا و مشتق خروجی فیلتر گیرنده ناهمبسته[۳۱] باشند.

نمودار بلوکی این روش را در زیر می‌بینیم.



شکل ۲ – بازیابی زمان decision-directed با روش MMSE

[30] Minimum Mean-Square-Error
[31] Uncorrelated

تنها نکته‌ای که به نظر مهم می‌رسد، این است که عمل جمع را به کمک یک فیلتر پایین‌گذر تحقق بخشیدیم. در واقع هر دوی آنها سیگنال را *هموار*[۳۲] می‌کنند؛ فیلتر پایین‌گذر فرکانس-های بالای سیگنال را از بین می‌برد و عمل جمع نیز با میانگین‌گیری روی سیگنال، آن را نرم تر خواهد کرد [۵].

خروجی VCO نیز هر دو نمونه‌بردار را کنترل خواهد کرد.

۴-۱-۲- روش بیشینه شباهت (*ML*)

همانطور که در بخش ۲ نشان دادیم، تابع لگاریتم شباهت به این شکل است (چون فقط قصد تخمین آفست زمانی τ را داریم، به جای $\boldsymbol{\theta}$ از τ استفاده کردیم):

$$\Lambda(\tau_0) = \frac{2}{N_0}\int_{T_0} r(t)s(t;\tau_0)\,dt$$

حال سیگنال $s(t;\tau)$ را به صورت زیر در نظر می‌گیریم:

$$s(t;\tau_0) = \sum_m a_m g_R(t - mT - \tau_0)$$

که یک سیگنال PAM[۳۳] باندپایه است.

با جایگزینی داریم:

$$\Lambda(\tau_0) = \frac{2}{N_0}\sum_m a_m \int_{T_0} r(t)g_R(t - mT - \tau_0)\,dt = \frac{2}{N_0}\sum_m a_m y_m(\tau_0)$$

که در این رابطه، $y_m(\tau_0)$ خروجی نمونه‌برداری شده‌ی فیلتر گیرنده است (مشابه آنچه در قبل بود).

$$y_m(\tau_0) = \int_{T_0} r(t)g_R(t - mT - \tau_0)\,dt$$

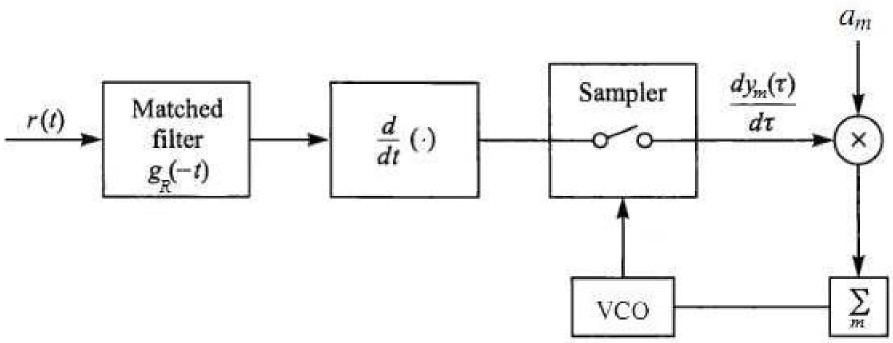
همانطور که مشاهده می‌شود، $\Lambda(\tau_0)$ خروجی فیلتر منطبق است که روی تعدادی از سمبل‌ها میانگین گرفته شده است.

با مشتق‌گیری از تابع لگاریتم شباهت و صفر قرار دادن آن داریم:

---
[32] Smooth
[33] Pulse Amplitude Modulation

$$\frac{d\Lambda(\tau_0)}{d\tau_0} = \frac{2}{N_0}\sum_m a_m \frac{dy_m(\tau_0)}{d\tau_0} = 0$$

شمای بلوکی این روش نیز به این صورت خواهد بود:



شکل ۳ – بازیابی زمان decision-directed با روش ML

در ادامه به کمک معیار ML، الگوریتمی را برای حالت non-decision-directed ارائه می-دهیم.

۴-۲- بازسازی زمان ***non-decision-directed***

در این روش نیازی به دانستن اطلاعات ارسالی نداریم و اساساً در مخابرات نیز، مسأله همین است. مطابق معمول تخمین ML را در این حالت بررسی می‌کنیم و با تغییر کوچکی در تابع لگاریتم شباهت، به یک مدار جدید می‌رسیم. سپس با بررسی کامل الگوریتم Early-Late Gate و ارائه-ی یک رابطه‌ی بازگشتی (فیلتر وفقی[۳۴]) سعی در دنبال کردن آفست زمانی خواهیم داشت. در نهایت به بیان مختصری از الگوریتم Gardner می‌پردازیم.

۴-۲-۱- روش بیشینه شباهت (***ML***)

در حالت non-decision-directed باید از تابع شباهت روی PDF سمبل‌های اطلاعات، میانگین گرفت و سپس اقدام به مشتق‌گیری کرد. اثبات می‌شود در حالتی که PDF را گوسی فرض کنیم، با تقریب خوبی قانون مربع[۳۵] (توان دوم) برقرار است. یعنی در رابطه‌ای که در

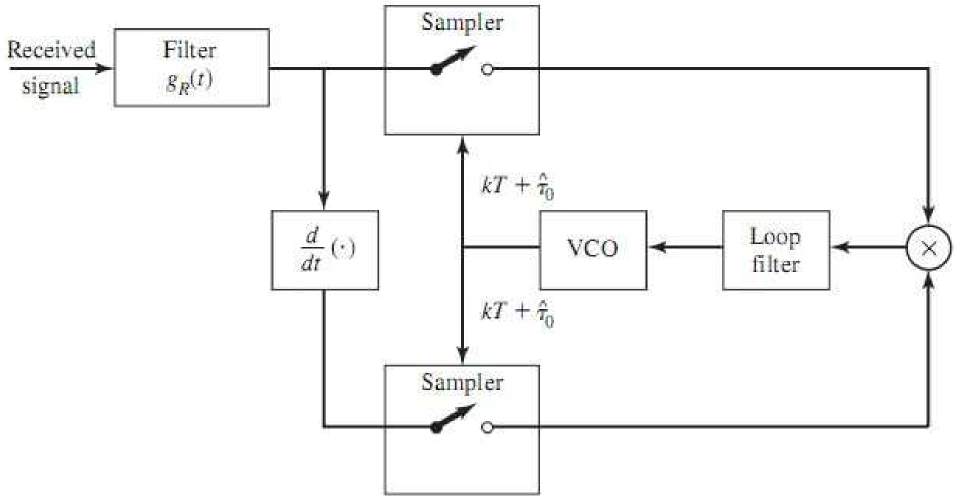
[34] Adaptive Filter
[35] Square Law

قسمت قبل بدست آوردیم، می‌توانیم به جای a_m از خود $y_m(\tau_0)$ استفاده کنیم تا رابطه‌ی لگاریتم شباهت این‌گونه بازنویسی شود:

$$\bar{\Lambda}(\tau_0) = \frac{2}{N_0}\sum_m y_m^2(\tau_0)$$

مشتق‌گیری از این رابطه نمودار بلوکی مطلوب را می‌دهد:

$$\frac{d\bar{\Lambda}(\tau_0)}{d\tau_0} = 2\sum_m y_m(\tau_0)\frac{dy_m(\tau_0)}{d\tau_0} = 0$$

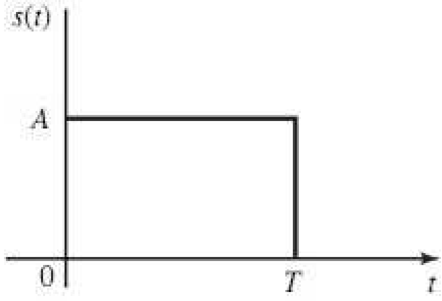


شکل ۴ – بازیابی زمان non-decision-directed با روش ML

البته می‌توانستیم ابتدا به کمک یک المان غیرخطی، سیگنال را به توان دو برسانیم و سپس مشتق بگیریم [۱].

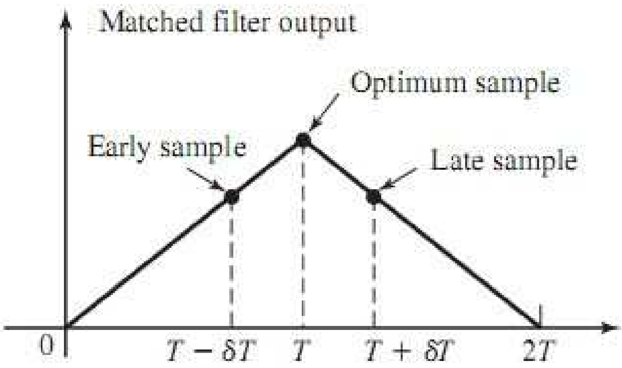
۴-۲-۲- همزمان‌ساز *Early-Late Gate*

همانطور که می‌دانیم، خروجی فیلتر منطبق حول نقطه‌ای که بیشینه مقدار خود را اختیار می‌کند، تقارن زوج دارد و این مسأله مستقل از این است که ورودی‌اش چه باشد. برای روشن‌تر شدن قضیه، فرض کنید که سیگنال پالس مستطیلی محدود در زمان زیر ورودی فیلتر منطبق باشد.



شکل ۵ – یک پالس مستطیلی به عنوان ورودی فیلتر منطبق

خروجی فیلتری که به این سیگنال منطبق باشد، به نوعی تابع خودهمبستگی آن خواهد بود که بیشینه خود را در $t = T$ اختیار می‌کند.

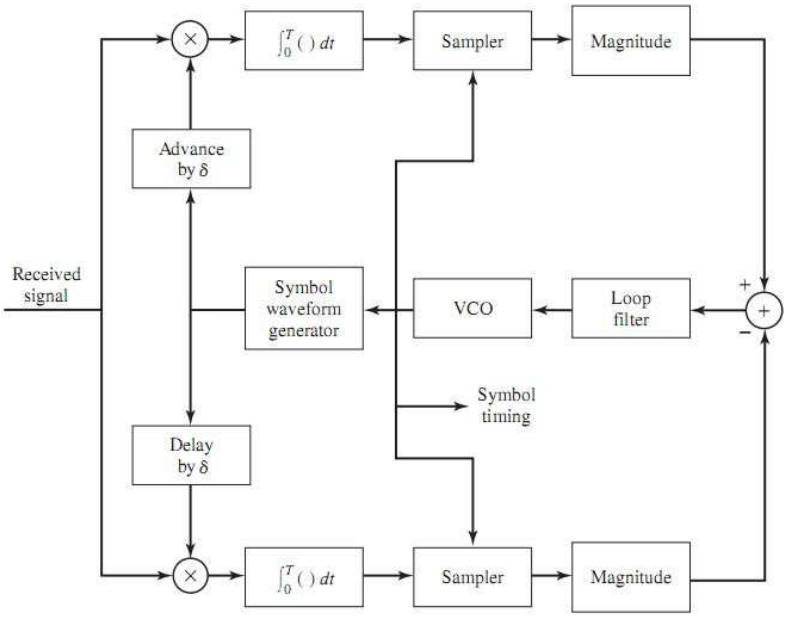


شکل ۶ – خروجی فیلتر منطبق به ورودی پالس مستطیلی

حال فرض کنید به جای نمونه‌برداری از سیگنال در $t = T$، یک لحظه قبلتر[۳۶] در $t = T - \delta T$ و یک لحظه بعدتر[۳۷] در $t = T + \delta T$ نمونه‌برداری می‌کنیم. با توجه به تقارن زوج، اندازه‌ی این دو نمونه در حالت ایده‌آل با هم برابر است. با این اوصاف زمان درست نمونه‌برداری، وسط بازه‌ی زمانی $(T - \delta T, T + \delta T)$ می‌باشد. در حضور نویز، قضیه قدری پیچیده می‌شود؛ دیگر لزوماً نمونه‌ها در لحظات قبل و بعد از $t = T$ با هم برابر نیستند. پس باید الگوریتمی تشکیل دهیم که سعی در برابر کردن این دو نمونه داشته باشد. یعنی اگر اندازه‌ی نمونه‌ی قبلی بزرگتر از اندازه‌ی نمونه‌ی بعدی باشد، خروجی VCO را افزایش می‌دهیم و در حالت دیگر نیز برعکس! این کار را تا زمانی ادامه می‌دهیم که هر دو نمونه برابر شوند.

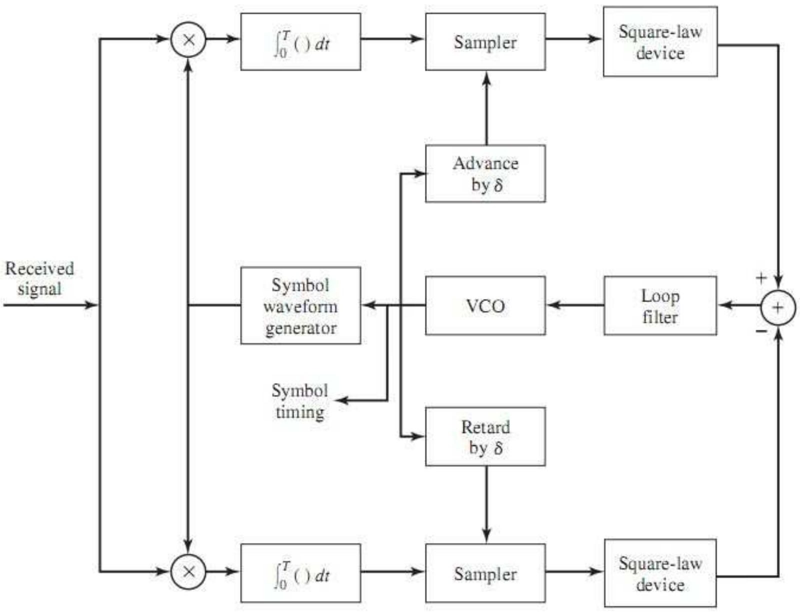
[36] Earlier
[37] Later

پیاده‌سازی الگوریتم به صورت زیر میسر است:



شکل ۷ – بازیابی زمان non-decision-directed با روش ELG – نوع اول

البته برای حالتی که میانگین دنباله‌ی سمبل‌های اطلاعات صفر باشد (مانند PAM) می‌توان از نمودار بلوکی زیر، که ساده‌تر نیز است، استفاده کرد.



شکل ۸ – بازیابی زمان non-decision-directed با روش ELG – نوع دوم

حال می‌خواهیم معیار دیگری را مطرح نماییم که نتیجه‌ی آن یک الگوریتم بازگشتی خواهد شد که می‌توانیم به کمک آن آفست زمانی را دنبال کنیم. خواهیم دید که نتیجه‌ی این الگوریتم، همان ELG خواهد بود.

معیاری که می‌خواهیم در نظر بگیریم، بیشینه کردن توان خروجی فیلتر گیرنده است.

۴-۲-۲-۱- بیشینه کردن توان خروجی

همانطور که قبلاً نیز به آن اشاره شد، در پی یافتن یک معادله‌ی بازگشتی برای یافتن پارامتر آفست زمانی τ هستیم. یک معیار مناسب می‌تواند بیشینه کردن متوسط توان دریافتی $E\{y^2[k]\}$ باشد. خروجی این الگوریتم τ^* باید به گونه ای باشد که با نمونه-برداری از سیگنال در زمان‌های $kT+\tau^*$، توان خروجی فیلتر گیرنده بیشینه شود.

تابع هدف ما این چنین خواهد شد:

$$J_{\mathrm{OP}}(\tau)=E\{y^2[k]\}=E\{y^2(kT+\tau)\}$$

حال به کمک الگوریتم steepest descent، $\tau[k]$ را اینگونه را روز می‌کنیم که اگر شیب $J_{\mathrm{OP}}(\tau)$ مثبت بود، $\tau[k]$ را زیاد و اگر منفی بود $\tau[k]$ را کم می‌کنیم[۳۸] [۶].

ساختار کلی الگوریتم به این صورت است:

$$\tau[k+1]=\tau[k]+\mu\left.\frac{dJ_{\mathrm{OP}}(\tau)}{d\tau}\right|_{\tau=\tau[k]}$$

برای محاسبه‌ی $\frac{dJ_{OP}(\tau)}{d\tau}$، می‌توان با تقریب خوبی جای عملگر مشتق و میانگین‌گیر را جا به جا کرد [۴]:

$$\frac{dJ_{\mathrm{OP}}(\tau)}{d\tau}\approx E\left\{\frac{dy^2[k]}{d\tau}\right\}=2E\{y[k]\frac{dy[k]}{d\tau}\}$$

و برای مشتق $y[k]$ نیز می‌توانیم از تقریب ابتدایی زیر، برای δ کوچک، استفاده کنیم:

$$\frac{dy[k]}{d\tau}=\frac{dy(kT+\tau)}{d\tau}\approx\frac{y(kT+\tau+\delta)-y(kT+\tau-\delta)}{2\delta}$$

حال با جایگزینی روابط داریم:

---

[۳۸] در الگوریتم‌های وفقی، معمولاً ازعلامت منفی برای ضریب μ استفاده می‌کنند، چرا که در آنجا هدف کمینه کردن تابعی از خطاست؛ اما اینجا می‌خواهیم توان را بیشینه کنیم!

$$\tau[k+1] = \tau[k] + \mu E\{y[k] \times (y(kT + \tau[k] + \delta) - y(kT + \tau[k] - \delta))\}$$

که δ را در μ جذب کردیم.

مرسوم است که عملگر $E\{.\}$ را حذف کرده و رابطه‌ی وفقی نهایی را اینگونه بنویسیم:

$$\tau[k+1] = \tau[k] + \mu y[k](y(kT + \tau[k] + \delta) - y(kT + \tau[k] - \delta))$$

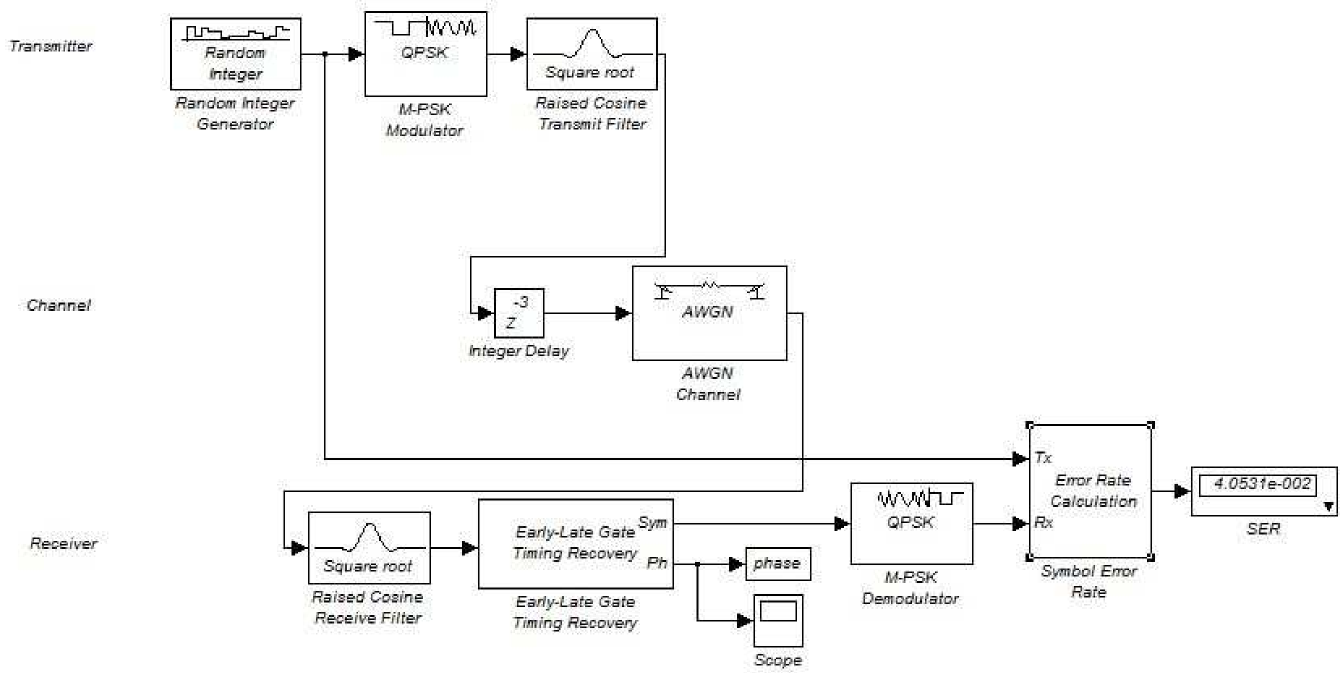
با تغییر μ می‌توان مصالحه‌ای بین سرعت و دقت همگرایی به وجود آورد.

همانطور که مشاهده می‌شود، نتیجه‌ای که در بالا بدست آمد، کاملاً بر مبنای الگوریتم ELG است و می‌توانیم شبیه‌سازی‌های خود را بر مبنای این الگوریتم انجام دهیم.

آنچه در انتهای این بخش می‌آید، دو شبیه‌سازی بر مبنای الگوریتم ELG خواهد بود.

۴-۲-۲-۲- شبیه‌سازی‌ها

شبیه‌سازی اول در محیط Simulink و به کمک بلوک Early-Late Gate Timing Recovery انجام شده است که شمایی از آن در زیر آمده است.



شکل ۹ – شبیه‌سازی روش ELG با Simulink

همانطور که در شکل نیز واضح است، داده‌های تصادفی در فرستنده تولید می‌شوند و پس از مدوله شدن (QPSK) و گذر از فیلتر Root Raised-Cosine (RRC) فرستنده، وارد کانال شده و در گیرنده نیز پس از عبور از فیلتر RRC وارد بلوک الگوریتم ELG شده و سپس عمل دمدولاسیون انجام می‌گیرد (به جای استفاده از یک فیلتر Raised-Cosine(RC) از دو فیلتر RRC بهره بردیم که علت را می‌توانید در [۴] جستجو کنید). در نهایت برای سنجش کیفیت کار، نرخ خطای سمبل (SER[۳۹]) نیز محاسبه می‌شود.

نتیجه‌ای که بدست آمد حدود 0.04 می‌باشد که چندان احتمال خطای خوبی نیست، اما قابل قبول است.

[39] Symbol Error Rate

شبیه‌سازی دوم به صورت M-File می‌باشد که فایل script آن را در ادامه بحث می‌توانید مشاهده کنید. طرز کار برنامه نیز به این صورت است که ابتدا یک سری اعداد تصادفی به عنوان داده‌های اولیه تولید می‌شوند (a). سپس یک تابع RRC تعریف کردیم که نقش شکل‌دهندگی پالس را دارد (g). این داده‌ها پس از upsample شدن، از g می‌گذرند و باید وارد کانال شوند (x). در کانال پس از عبور از یک فیلتر (فعلاً در اینجا Butterworth) نویز سفید گوسی به آن اضافه می‌شود (چند نوع فیلتر طراحی شده‌اند). در گیرنده نیز ابتدا داده‌های رسیده را از فیلتر RRC عبور می‌دهیم و سپس با ایجاد یک تأخیر تصادفی، داده‌ها را به روز می‌کنیم (r). پس تا اینجا ما اطلاعات را با یک تأخیر نامشخص دریافت کردیم و حال می‌خواهیم این تأخیر را به کمک خود داده‌ها (non-decision-directed یا non-data-aided) تخمین بزنیم.

الگوریتم ما با محاسبه‌ی $y[k]$، $y(kT+\tau[k]+\delta)$ و $y(kT+\tau[k]-\delta)$ به ترتیب در زمان‌های tkr، tkr_early و tkr_late آفست زمانی تخمین‌زده شده (tau_est) را به کمک رابطه‌ی وفقی بدست آمده در بخش قبل روز می‌کند. مقدار اولیه برای tau_est نیز صفر در نظر گرفته شده است. برای محاسبه‌ی $y[k]$ ها، از فیلتر منطبق استفاده شده (یک فیلتر RC) که آنرا با تابع دیگری تعریف کردیم ($interpolate$). کد این تابع را نیز در ادامه آمده است.

در نهایت باید به این نکته اشاره کنیم که نرخ سمبل $f = 2.4^{MHz}$ می‌باشد و هر سمبل شامل 10 نمونه می‌باشد. شبیه‌سازی برای 10000 سمبل و $\mathrm{SNR} = 20^{\mathrm{dB}}$ اجرا شده است.

کد های MATLAB

$\% \ Constructs \ an \ RC \ filter, with \ zeros \ at \ kT_s \ with \ rolloff \ factor \ r \ and \ delay \ \mu T_s \ \%$

$function \ g \ = \ interpolate(M, r, mu)$
$f \ = \ 10\hat{}(-7);$
$t1 \ = \ sin(pi * ((-M{:}M) + mu) + f) \ ./ \ (pi * ((-M{:}M) + mu) + f);$
$t2 \ = \ cos(pi * r * ((-M{:}M) + mu) + f) \ ./ \ (1 - (2 * r * (-M{:}M) + mu) + f).\hat{}2;$
$g(:,1) \ = \ t1 \ .* \ t2;$

$\% \ Constructs \ an \ RRC \ pulse, with \ zeros \ at \ kT_s \ with \ rolloff \ r \ and \ delay \ \mu T_s \ \%$

$function \ g \ = \ interpolate2(M, r, mu)$
$f \ = \ 10\hat{}(-7); \ T = 1;$
$t1 \ = \ cos((1 + r) * pi * ((-M{:}M) + mu) + f);$
$t2 \ = \ sin((1 - r) * pi * ((-M{:}M) + mu) + f)./(4 * r * ((-M{:}M) + mu) + f);$
$t3 \ = \ pi * sqrt(T) * (1 - (4 * r * ((-M{:}M) + mu) + f).\hat{}2);$

$g(:,1) \ = \ 4 * r * (t1 + t2)./t3;$

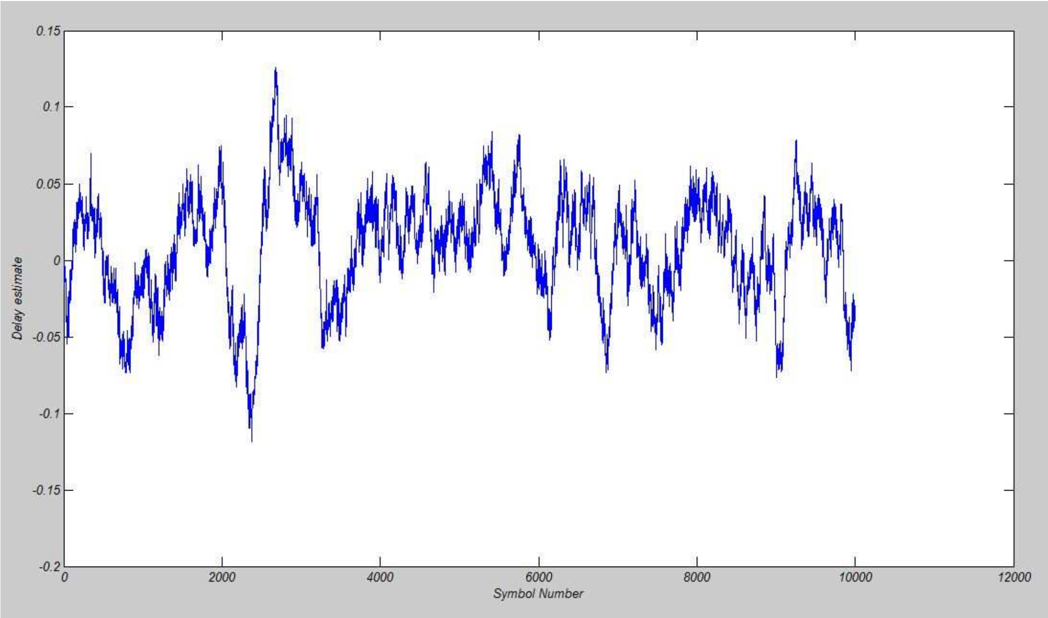
```
% Timing Recovery via Output Power Maximization, Early − Late Gate algorithm %

clear all
clc

% Setting variables %

f = 2.4e6; T = 1/f;          % Symbol frequency/period
fs = 10 * f; Ts = 1/fs;      % Sampling frequency/period (T/T_s = 10)
fc = 6e6;                    % Carrier frequency
N = 10000;                   % Number of symbols
L = 3;                       % Length of total pulse is 2L(T/T_s) + 1
M = 8;                       % Length of interpolation filter is 2M + 1
mu = 0.02;                   % Algorithm stepsize
delta = 0.01;                % Differentation step
rolloff = 0.6;               % Roll_off factor for pulse shaping
rolloff2 = 0.5;              % Roll_off factor for interpolateolation

% Construction of pulse train and channel %

a = sign(randn(N,1));                              % Input data
g = rcosine(1/T, 1/Ts, 'fir/sqrt', rolloff, L);    % Symbol pulse shape
x = conv(upsample(a, 2
          * T/Ts), g);        % Upsampling and then passing through RRC filter

% Designing different types of filters %

Hd1 = dfilt.allpass([1.5 0.7]);                          %Allpass filter

Hd2 = design(fdesign.bandpass(1e6, 3e6, 9e6, 11e6, 60, 1, 80, 24e6),
          'ellip', 'MatchExactly', 'both');              %Bandpass filter

Hd3 = design(fdesign.lowpass(8e6, 10e6, 1, 80, 24e6),
          'ellip', 'MatchExactly', 'both');              %Lowpass filter

HdLP = design(fdesign.lowpass('N, F3dB', 15, 1.97e6, 24e6), 'butter');
x = filter(HdLP, x);
x = awgn(x, 20);                                         % Adding noise


% Receiver %
x = conv(x, g);                             % RRC filter at receiver
```

```
Lx = length(x);

% Random timing offset (supposed to be the effect of channel)
tau = floor(10 * randn);

if (tau > 0)
r = [zeros(tau,1); x(1:Lx - tau)];          % Constructing the receiving data
else
r = [x(-tau + 1:Lx); zeros(-tau,1)];
end
 % Algorithm for estimating tau %

tau_est(1) = 0;                              % Initial value

for k = 1:N

tk = k * T + tau_est(k) * T + Ts;
tkr = round(tk/Ts);
muk = (tk - tkr * Ts)/Ts;
h(:,1) = interpolate(M,rolloff2,muk);
interpolate_x(k) = h.'* r(tkr + M:-1:tkr - M);          % Matched filtering

% Compute of "early" parameters %

tk_early = k * T + tau_est(k) * T - delta * T + Ts;          % A little bit earlier
tkr_early = round(tk_early/Ts);
muk_early = (tk_early - tkr_early * Ts)/Ts;
h_early(:,1) = interpolate(M,rolloff2,muk_early);
interpolate_x_early = h_early.' * r(tkr_early + M:-1:tkr_early - M);

% Compute "late" parameters %

tk_late = k * T + tau_est(k) * T + delta * T + Ts;          % A little bit late
tkr_late = round(tk_late/Ts);
muk_late = (tk_late - tkr_late * Ts)/Ts;
h_late(:,1) = interpolate(M,rolloff2,muk_late);
interpolate_x_late = h_late.' * r(tkr_late + M:-1:tkr_late - M);

% Adaptation iteration %

tau_est(k + 1) = tau_est(k) + mu * interpolate_x(k) *
                 (interpolate_x_late - interpolate_x_early);
end

% Plotting the results
```

```
tau_est = 10 * tau_est;
plot(tau_est)
xlabel('Symbol Number')
ylabel('Delay estimate')

% 500 is an estimate of the number of symbol for the algorithm to converge
avg = mean(tau_est(500:N));
tau_est = round(avg)                    % Printing the result
```

نتایج شبیه‌سازی‌ها بعد از حدوداً 400 سمبل به مقدار درست τ همگرا می‌شود.

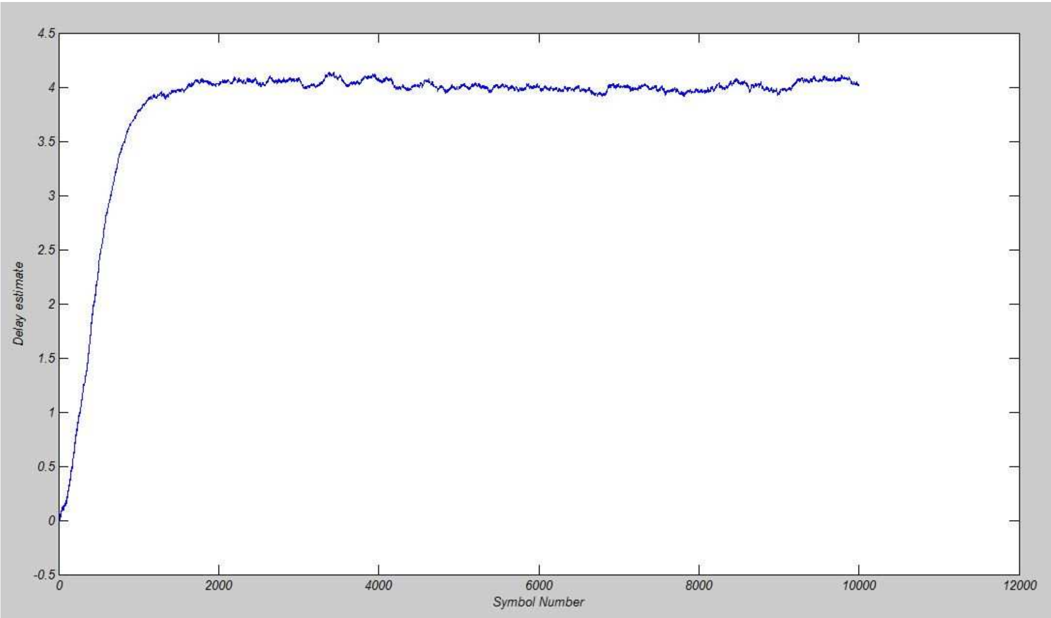
در حالتی که τ در گیرنده را 0 در نظر بگیریم، τ_{est} مقدار τ را با روند زیر دنبال می‌کند:



شکل ۱۰ – دنبال کردن آفست زمانی برای $\tau = 0$

میانگین این مقادیر 0.0067 است که بسیار به صفر نزدیک است. شاید برای $\tau = 0$ نتوان به خوبی خاصیت دنبال‌کنندگی الگوریتم را دید؛ مثال زیر قضیه را روشن‌تر می‌کند.

فرض کنید $\tau = 4$ باشد. نتیجه به این صورت است:



شکل ۱۱ – دنبال کردن آفست زمانی برای $\tau = 4$

این بار میانگین این مقادیر 3.9681 است که باز هم حاکی از صحت کار الگوریتم دارد و همانطور که مشاهده می‌شود الگوریتم به خوبی مقدار τ را دنبال می‌کند.

۴-۲-۳- الگوریتم ***Gardner***

همان‌طور که پیش از این نیز بیان شد، هدف اصلی پروژه بررسی کامل الگوریتم ELG بود که آن را بررسی کردیم. حال برای کامل‌تر کردن موضوع، الگوریتم معروف دیگر بازیابی ساعت را نیز معرفی می‌کنیم.

این الگوریتم به صورت عمده مورد استفاده قرار می‌گیرد و در هر سمبل به دو نمونه نیاز است. همچنین به دلیل عدم حساسیت نسبت به آفست فاز حامل، می‌تواند قبل از بازیابی حامل به کار گرفته شود.

خطای الگوریتم نیز توسط رابطه‌ی زیر به روز می‌شود:

$$e_n = (y_n - y_{n-2})y_{n-1}$$

که در این رابطه فاصله‌ی بین y_n و y_{n-2}، T می‌باشد و فاصله‌ی بین y_n و y_{n-1} نیز $T/2$ خواهد بود (T دوره‌ی یک سمبل و y_n نیز نمونه در لحظه‌ی فعلی است).

برای مثال اگر نقاط نمونه‌برداری شده در لحظات درست 1، 0 و -1 باشند، خطا صفر خواهد بود. اما اگر نقاط نمونه‌برداری شده، اشتباهاً 0.9، -0.1 و -0.9 باشند (یعنی قدری جابه-جایی به راست)، آنگاه خطا 0.18 خواهد شد که نشان می‌دهد با در جهت کاهش خطا، زمان نمونه‌برداری را قدری به چپ شیفت دهیم. به طور مشابه برای خطای منفی نیز می‌توان استدلال کرد.

۵-نتیجه‌گیری

همزمان‌سازی سمبل (بازیابی زمان) یکی از مهمترین قسمت‌های یک سیستم مخابراتی به حساب می-آید. برای این مهم، الگوریتم‌های متعددی نیز ارائه شده‌اند که ما در این پروژه به الگوریتم مشهور Early-Late Gate پرداختیم و با انجام شبیه‌سازی‌ها نشان دادیم که با معیار بیشینه توان خروجی، می‌توان آفست زمانی مورد نظر را تخمین زد و دنبال کرد. در فرآیند تخمین، مصالحه‌ای بین سرعت و دقت همگرایی وجود دارد که با تغییر آن می‌توان به دقت یا سرعت مطلوب رسید. الگوریتم ارائه شده نسبت به نویز هم مقاومت خوبی از خود نشان می‌دهد.

۶- مراجع

[1] J. Proakis, M. Salehi, *Digital Communications*, McGraw-Hill, 2008
[2] L. E. Franks, “Carrier and bit synchronization in data communication – A tutorial review,” *IEEE Trans. Commun.*, vol. COM-28, no. 8, pp. 1107-1120, Aug. 1980
[3] H. L. Van Trees, *Detection, Estimation, and Modulation Theory, Part I*, New York, Wiley, 1968
[4] C. R. Johnson Jr., W. A. Sethares, *Telecommunication Breakdown*, Prentice-Hall, 2003
[5] J. Proakis, M. Salehi, *Communication Systems Enginnering*, Prentice-Hall, 2002
[6] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, 1986

Abstract

One of the most crucial parts of a communication system is synchronization, which consists of two major sections, namely timing recovery and carrier recovery. In this project, after some discussions on the theory of parameter estimation and matched filters, we elaborate the methods used in timing recovery and with the needed simulations, we will try to make up for the timing offset.

University of Tehran

Faculty of Engineering

Department of Electrical and Computer Engineering

Thesis Title:

Simulation and Comparison between Different Methods of Symbol Synchronization and Tracking them for Linear Modulation Schemes

By:

Morteza Banagar

Advisor:

Dr. Ali Olfat

Judge:

Dr. Maryam Sabbaghian

A thesis submitted to the undergraduate studies office in partial fulfillment of the requirements for the degree of B.Sc. in Electrical Engineering

September 2012