

【レビンソン・ダービンアルゴリズム Levinson-Durbin Algorithm】

線形予測係数 $\alpha_m (m = 1, 2, \dots, M)$, 自己相関関数 $r_{xx}(i) (i = 0, 1, \dots, M)$, PARCOR 係数 k_m

Initialization

$$\begin{aligned} \alpha_0^{(0)} &= 1 \\ \alpha_1^{(0)} &= 0 \\ w_0 &= r_{xx}(1) \\ u_0 &= r_{xx}(0) \\ m &= 1 \end{aligned}$$

step1.

$$\begin{aligned} k_m &= \frac{w_{m-1}}{u_{m-1}} \\ u_m &= u_{m-1}(1 - k_m^2) \end{aligned}$$

step2.

$$\begin{aligned} \alpha_i^{(m)} &= \alpha_i^{(m-1)} - k_m \alpha_{m-i}^{(m-1)} \quad (i = 1, 2, \dots, m) \\ \alpha_0^{(m)} &= 1 \\ \alpha_{m+1}^{(m)} &= 0 \end{aligned}$$

step3.

```
if (m==M)
    end
else
    go to step4
```

step4.

$$w_m = \sum_{i=0}^m \alpha_i^{(m)} r_{xx}(m+1-i)$$

step5.

```
m = m + 1
go to step1.
```

Source Code 1: Scilab

```
////////////////////////////////////
//      レビンソン・ダービンアルゴリズム
//      Levinson-Durbin Algorithm
//
//                                     M.Tsutsui
////////////////////////////////////

funcprot(0); //レビンソン・ダービンアルゴリズム関数
function[alpha]=my_lev_durb(order,rxx); //引数 order:次数 r:相関係数 auto_corr関数に関して,次数+1個必要
//-----初期値-----//
alpha=[]; //LPC係数
alpha_d=[]'; //LPC係数一時保持用
alpha_d(1)=1;
k=[]; //PARCOR係数
k(1)=0;

w=0;
u=[];

alpha(1)=1; //LPC係数 固定
```

```

alpha(2)=0;//LPC係数初期化

w(1)=rxx(2); //r(1)
u(1)=rxx(1); //r(0)

//-----m=1__must处理____//
k(1)=0;

k(2)=w(1)/u(1);
u(2)=u(1)*(1-k(2)^2);

alpha(2)=alpha(2)-k(2)*alpha(1);

alpha_d(2)=alpha(2);

w(2)=alpha(1)*rxx(3)+alpha(2)*rxx(2);//i=2

alpha(1)=1;
alpha(3)=0;

//-----m=2以上处理____//
for m=2:1:order+1;

    k(m+1)=w(m)/u(m);
    u(m+1)=u(m)*(1-k(m+1)^2);

    alpha(2)=alpha(2)-k(m+1)*alpha(m);

    alpha_d_buf=[];
    for o=1:1:m-1;
        alpha_d_buf=[alpha_d_buf,alpha_d(o)];
    end

    alpha_d_buf=flipdim(alpha_d_buf,2);//flipdim: 配列反転

    for i=1:1:m-1;
        alpha(i+2)=alpha(i+2)-k(m+1)*alpha_d_buf(i);
    end

    for j=2:1:m+1;
        alpha_d(j)=alpha(j);
    end

    alpha(1)=1;
    alpha(m+2)=0;

    alpha_buf=[];
    for r=1:1:m+1;
        alpha_buf=[alpha_buf,alpha(r)];
    end

    rxx_buf=[];
    for h=2:1:m+2;
        rxx_buf=[rxx_buf,rxx(h)];
    end

    w(m+1)=sum(alpha_buf.*flipdim(rxx_buf,2));

end

alpha=alpha(1:(length(alpha)-1));
endfunction

```