

信号与系统实验

基于通信系统的 被动感知

张子尚 徐建辉 刘子羽 华羽霄



目录

CATALOGUE

1 项目背景

2 研究内容

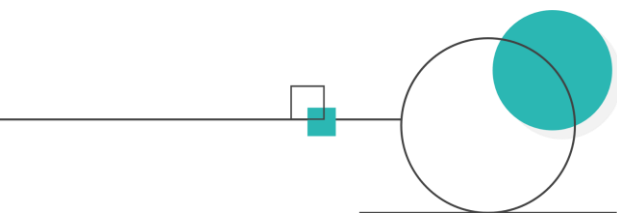
3 实验结果

4 拓展优化





项目背景



被动雷达基本原理

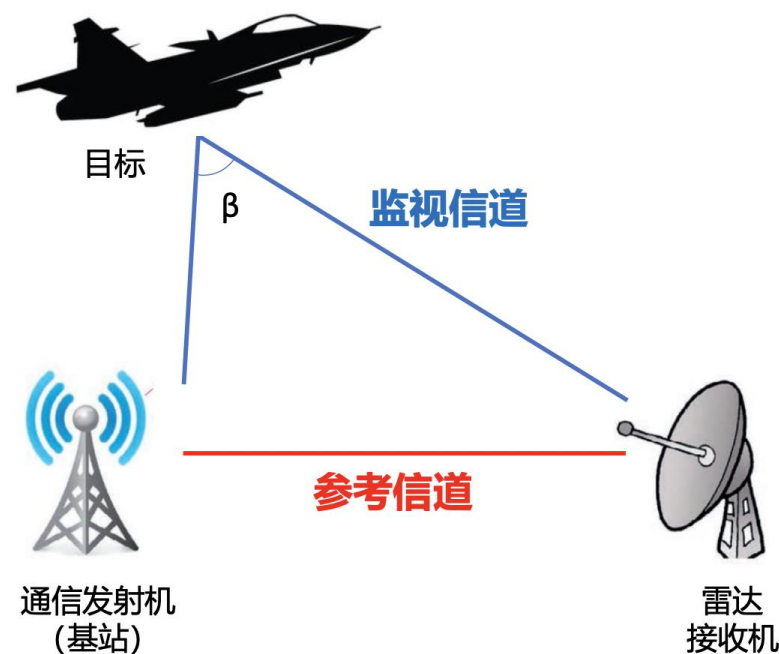
基本原理:

被动雷达首先接收一个从外部发射机发出, 经直达径(通常称为“**参考信道**”)传输的参考信号。同时, 它会收到同一信号经由目标散射(称为“**监视信道**”)后的回波信号。通过计算从两个信道收集的信号之间的时间差和频率差来估计目标距离和多普勒频移。

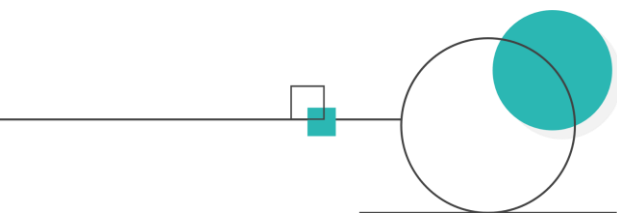
目标距离与速度的计算:

路径长度差 = 信道时延差 × 光速

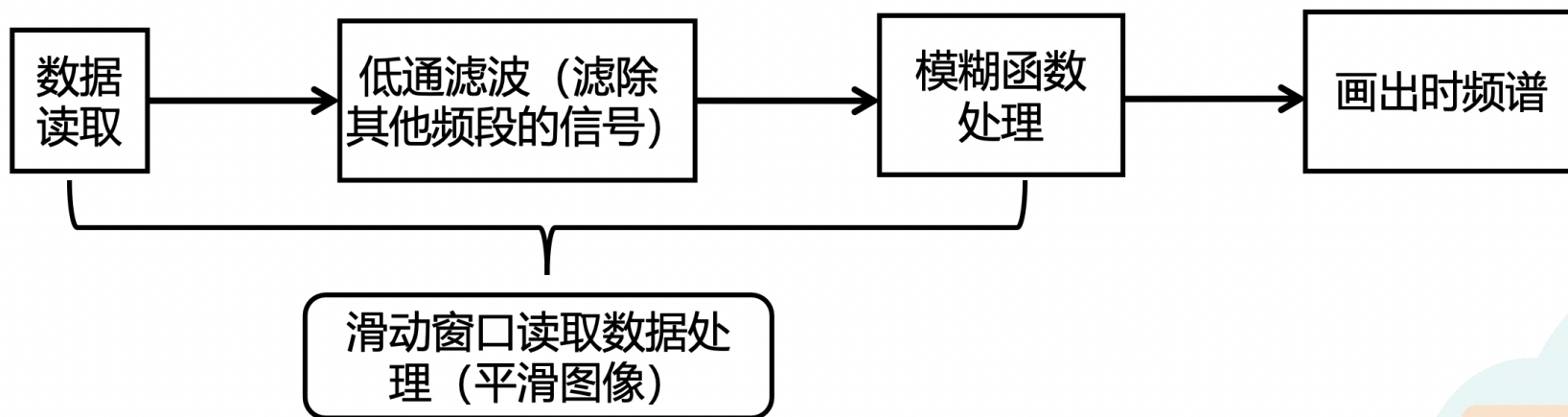
$$\text{目标速度} = \frac{\text{波长} \times \text{多普勒频移}}{2\cos(\beta)}$$



研究内容



信号处理流程



被动雷达基本原理

假设 $x(t)$ 为发射信号

参考信号 $y_{ref}(t) = \alpha x(t - \tau_r)$ 为时延和衰减后的发射信号

监视信号 $y_{sur}(t) = \beta x(t - \tau_s) e^{j2\pi f t}$ 为衰减、时延和多普勒频偏后的发射信号

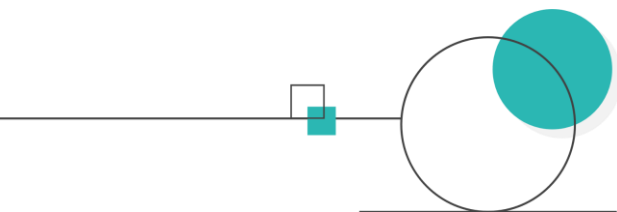
定义模糊函数

$$Cor(c, d) = \int_t^{t+T} y_{sur}(t + c) y_{ref}^*(t) e^{-j2\pi d t} dt$$

估计时延差和多普勒频移 (τ, f) :

$$(\hat{\tau}, \hat{f}) = \arg \max_{c, d} Cor(c, d)$$

实验结果

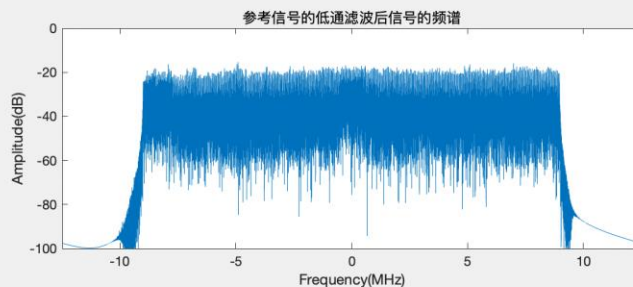
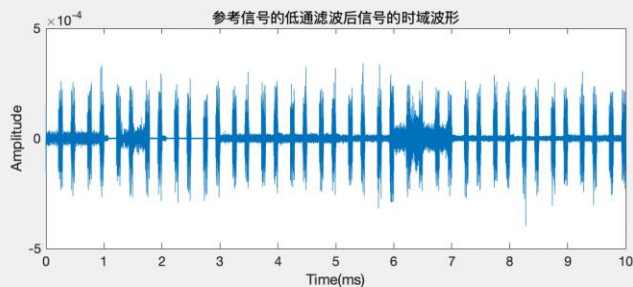
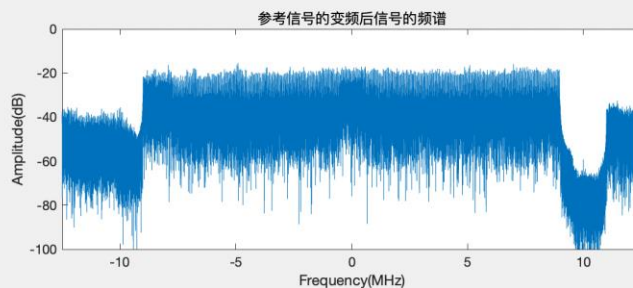
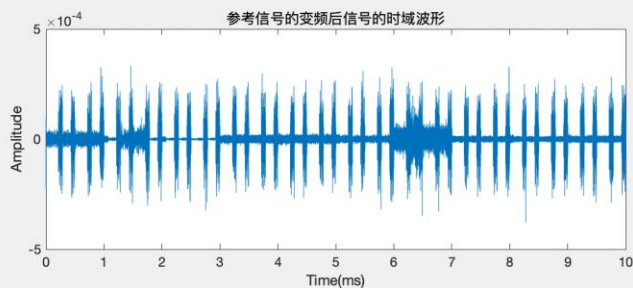
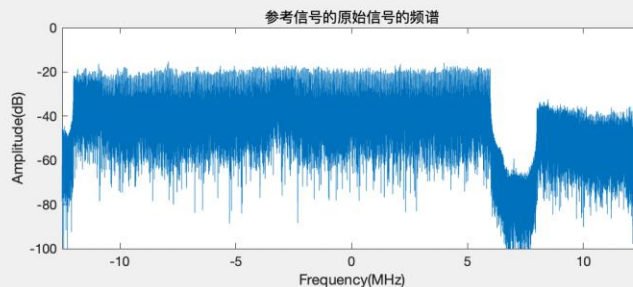
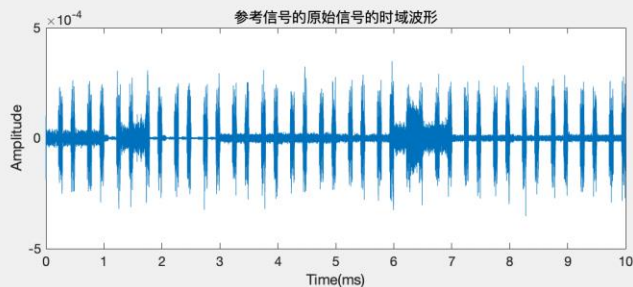


参考信号经过处理后的时域和频域图



SUSTech Southern University
of Science and
Technology

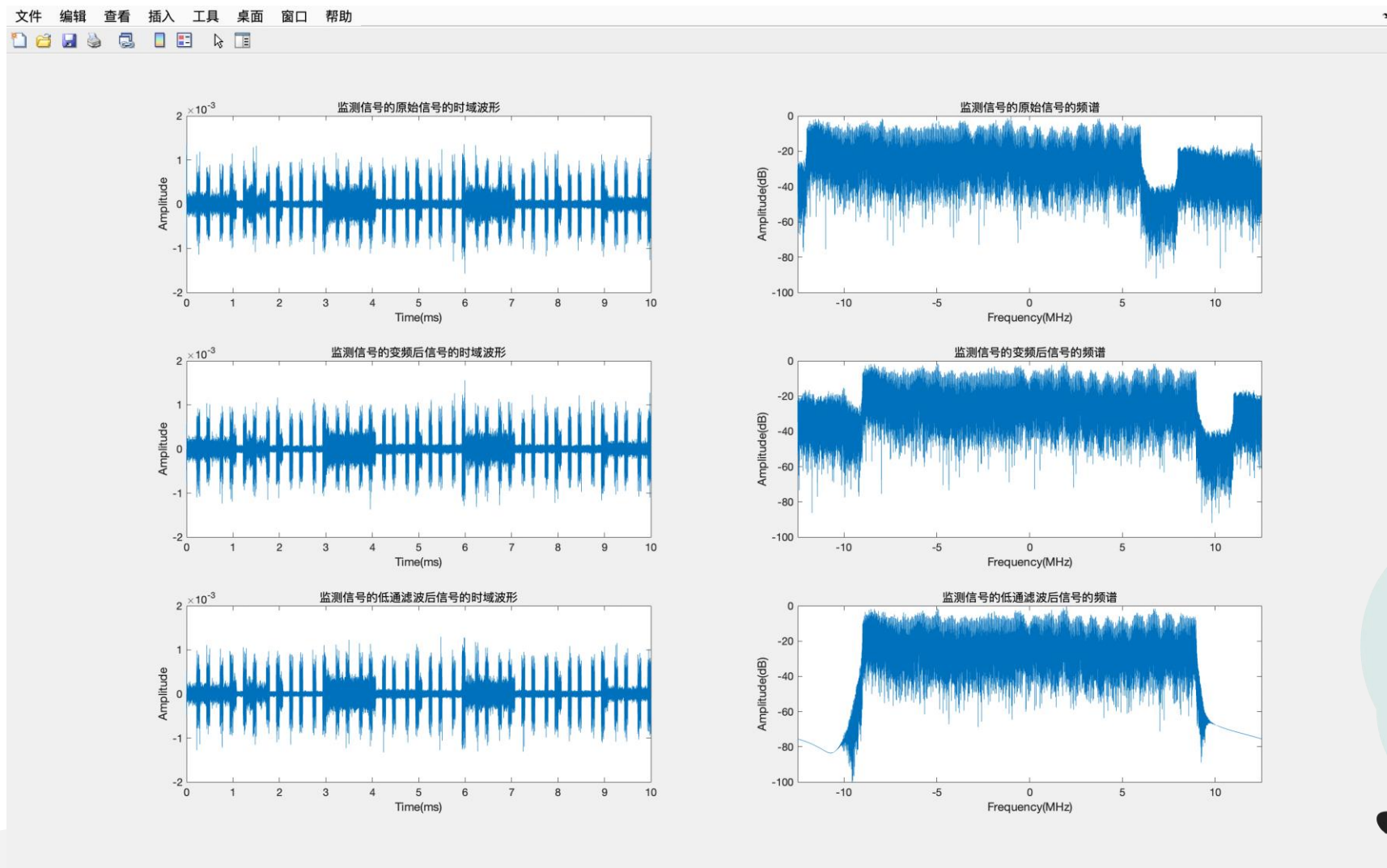
文件 编辑 查看 插入 工具 桌面 窗口 帮助



监测信号经过处理后的时域和频域图



SUSTech Southern University
of Science and
Technology

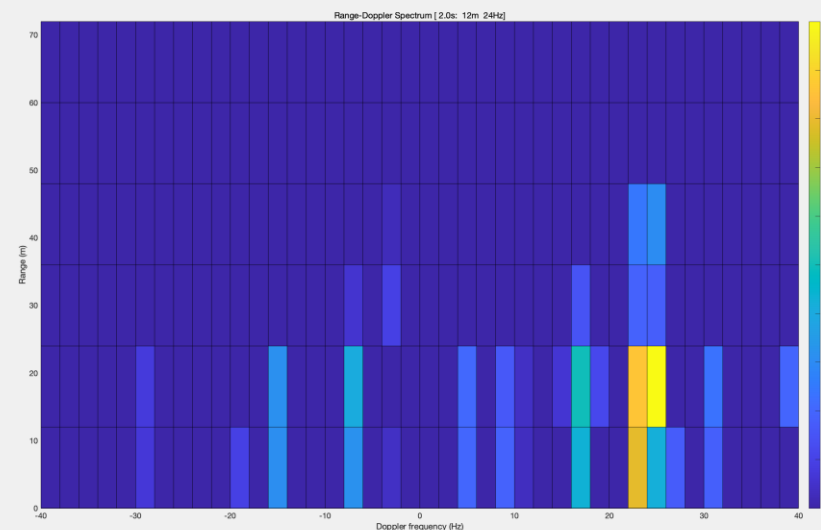
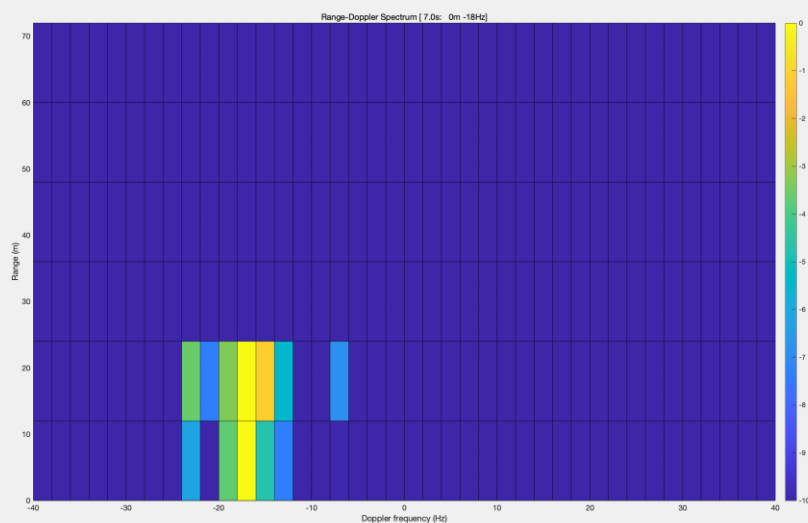
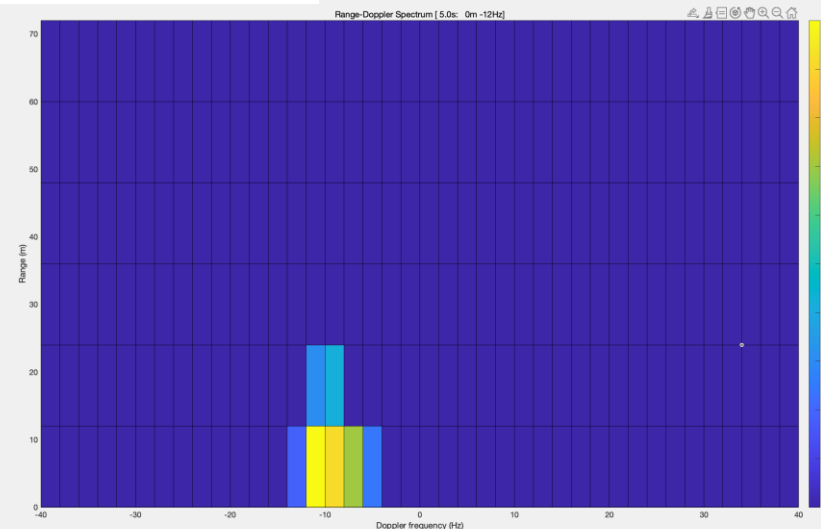
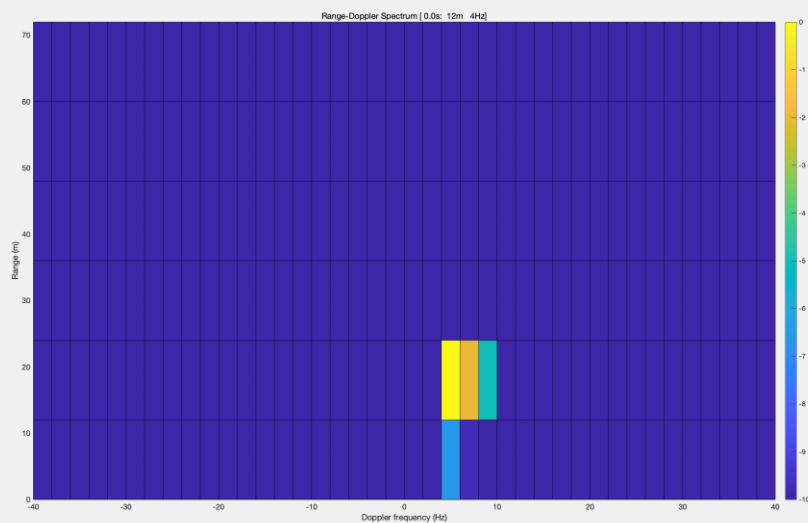


0~0.5s、2~2.5s、5~5.5s、7~7.5s 信号处理后得到的距离-多普勒谱

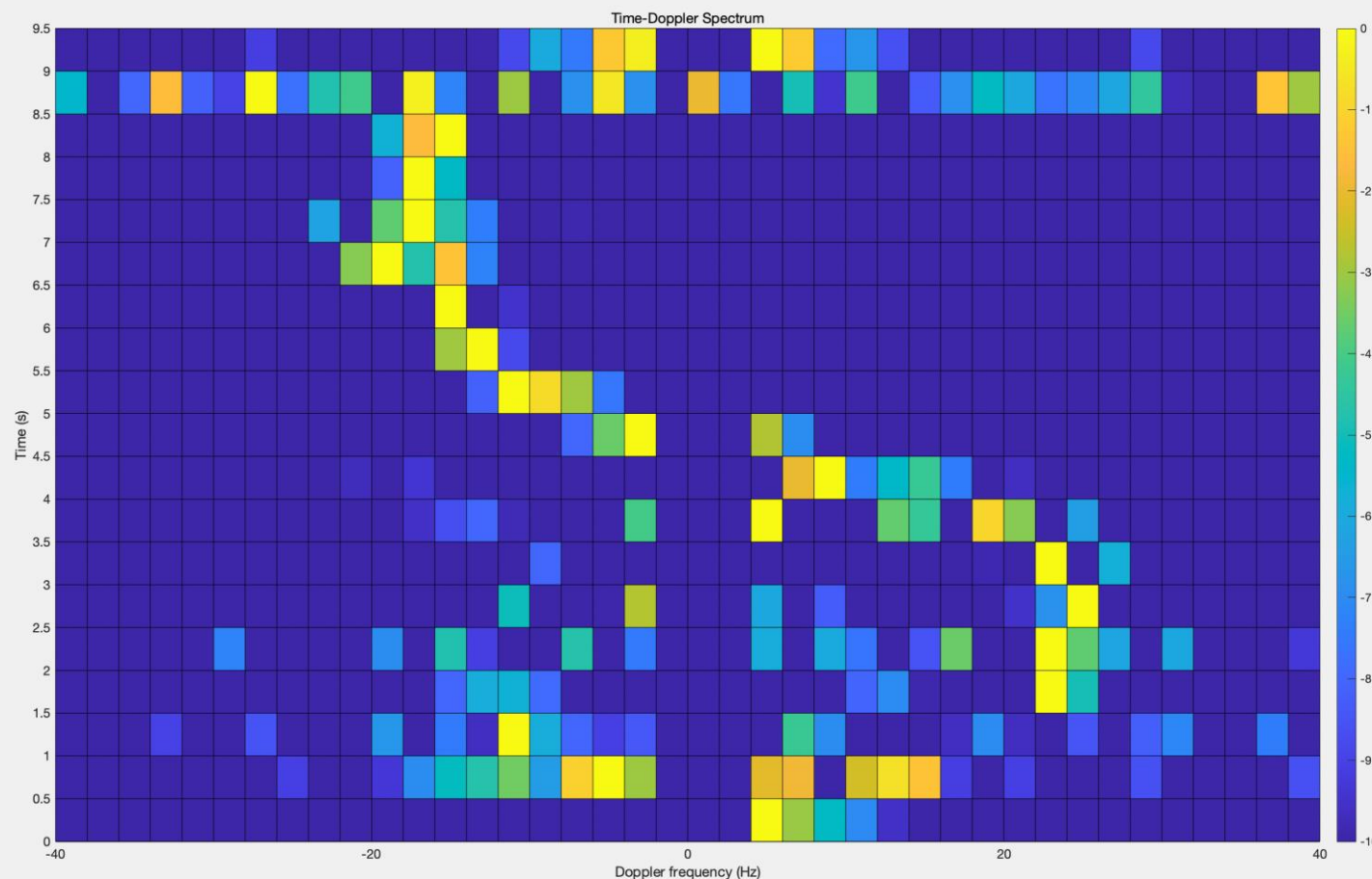


SUSTech

Southern University
of Science and
Technology



以CIT=0.5s为滑动窗口每次滑动0.5s 处理得到的10s数据的多普勒-时间谱



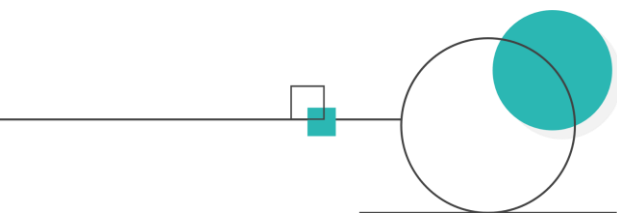
路径长度差 = 信道时延差 × 光速

$$\text{目标速度} = \frac{\text{波长} \times \text{多普勒频移}}{2\cos(\theta/2)}, \theta = 90$$

1. 从图可以看出，随着时间增长，模糊函数最大时的多普勒频移先增加再减小。
2. 根据公式可以得到，目标速度先从正方向加速到最大值，之后减速一直到负方向最大值，最终速度变为零



拓展优化



定义模糊函数


$$Cor(c, d) = \int_t^{t+T} y_{sur}(t + c) y_{ref}^*(t) e^{-j2\pi dt} dt$$

length(y_sur) == length(y_ref) == 12500000



方法1：朴素算法

```
xxxx=0;  
for c_i=c  
    for d_i=d  
        for k= 1:len  
            Back=0;  
            Back=y_sur(k+c_i)*y_ref_conj(k)*exp(-1i*2*pi*d_i*k)+Back;  
            xxxx=xxxx+1;  
            out(xxxx)=Back;  
        end  
    end  
end
```

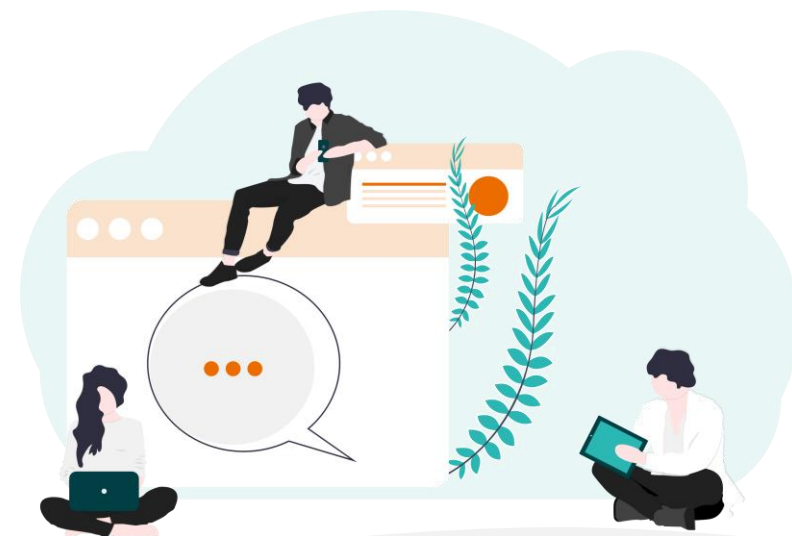


$\text{Back} = y_{\text{sur}}(k+c_i) \cdot y_{\text{ref_conj}}(k) \cdot \exp(-1i \cdot 2 \cdot \pi \cdot d_i \cdot k) + \text{Back};$

$$\text{Cor}(c, d) = \int_t^{t+T} y_{\text{sur}}(t + c) y_{\text{ref}}^*(t) e^{-j2\pi dt} dt$$

$\text{Count_time} > 1h$

最原始，人类传统计算方法



方法2：点乘优化

```
xxxx=0;  
max_out=0;  
for c_i=c  
    c_zero=zeros(1,c_i);  
    for d_i=d  
        y_sur_zero=[c_zero y_sur];  
        y_ref_conj_zero=[y_ref_conj c_zero];  
        exp_125_zero=[exp_125 c_zero];  
        xxxx=xxxx+1;  
        out(xxxx)=sum(y_sur_zero.*y_ref_conj_zero.*exp_125_zero);  
        if abs(out(xxxx))>max_out  
            c_argmax=c_i;  
            d_argmax=d_i;  
            max_out=abs(out(xxxx));  
        end  
    end  
end
```

`out(xxxx)=sum(y_sur_zero.*y_ref_conj_zero.*exp_125_zero);`

`count_time =`

76.0349

计算机硬件层面优化，加快向量读取速度，同时利用Matlab底层算法优化



方法3：快速傅里叶变换

```
cor_n=zeros(length(array_sample_shift),f_s/2);
for t=1:length(array_sample_shift)
    ser_zeros=[seq_sur_lpf(array_sample_shift(t)+1:end),zeros(1,array_sample_shift(t))];
    ref_=seq_ref_lpf;
    cor_n(t,:)=fftshift(fft(ser_zeros.*conj(ref_)));%得到cor 关于f的函数
end
A_TRD(idx_start_time,,:)=cor_n(:,(-19+f_s/4:21+f_s/4));
```

Complexity reduces to
 $O(N \log N)$
 $e^{-j2\pi dt}$

`cor_n(t,:)=fftshift(fft(ser_zeros.*conj(ref_)));%得到cor 关于f的函数`

count_time =

3.2902





感谢
您的观看和倾听