**摘要**

钻具信号的处理一直都是了解钻具在井下运行情况的重要措施，它能够让钻井工作人员实时掌握到钻具在井底的真实情况，预防安全事故的发生，因此，工作人员获得准确的钻具信号必须接近真实信号。为了获得准确的钻具信号，必须对钻具信号进行滤波处理，所以对钻具的信号进行滤波在钻具振动信号处理中是一个重要的环节，它能够使工作人员得到更加准确的钻具信号，清楚的了解钻具真实的运行情况。

本文介绍的基于Winform的钻具信号滤波器，采用Winform平台对钻具信号进行滤波处理，能够在现场工程中对数据进行滤波处理，为了实现基于Winform进行钻具信号的滤波处理，若使用以往常用的方式则无法完成在现场工程对数据进行滤波处理，所以本文采用MATLAB与Winform进行混合编程实现滤波器的设计，通过MATLAB对现场数据进行数据处理和频谱分析，然后生成在Winform平台可直接调用的dll与.h文件，然后在Winform平台中对dll与.h文件进行调用。在Winform平台中调用数据库后，通过调用dll与.h文件进行滤波处理，最后将滤波后数据显示到Winform界面。

在本文的研究中实现了对现场数据的频谱分析，算法对比以及优化滤波后滤波效果的比较，搭建了Winform显示平台，结合所选的滤波算法、数据分析及优化算法处理后在Winform平台上显示滤波后数据图，在经过滤波处理后得到了准确的钻具信号。整个论文介绍了在基于Winform平台的基础上完成了对钻具信号带通滤波器的滤波与显示界面的设计，能够对现场数据进行滤波处理得到准确的钻具信号。

关键字：钻具信号；Winform；滤波处理；混合编程；动态链接库调用

**Abstract**

The processing of the drilling signal has always been an important measure to understand the operation of the drill in the downhole. It allows the driller to grasp the real situation of the drill in the bottom of the well and prevent the occurrence of a safety accident. Therefore, the staff is accurate The drilling signal must be close to the real signal. In order to obtain accurate drilling signal,it is necessary to filter the drilling signal. Therefore, filtering the signal of the drilling tool is an important link in the drilling vibration signal processing,which can make the staff get more accurate drilling signal,A clear understanding of the real operation of drilling tools.

This paper introduces the Winform-based drilling signal filter,using Winform platform to filter the drilling signal processing,in the field engineering data filtering,in order to achieve based on Winform drilling tool signal filtering,if used in the past The method can not be completed in the field engineering data filtering, so this paper uses MATLAB and Winform mixed programming to achieve the design of the filter,through MATLAB on-site data for data processing and spectrum analysis, and then generated in the Winform platform can be directly called Dll and .h files, and then in the Winform platform on the dll and. H file to call. Call the database in the Winform platform,by calling dll and. H file for filtering,and finally the filtered data displayed to the Winform interface.

In this paper,we have realized the comparison of the spectral analysis,the comparison of the algorithm and the optimization of the filtering effect on the field data,and built the Winform display platform,which is displayed on the Winform platform after combining the selected filtering algorithm,data analysis and optimization algorithm After filtering the data,after the filtering process has been accurate drilling signal.The whole paper introduces the design of the filter and display interface of the bandpass filter of the drilling signal based on the Winform platform,and can filter the field data to get the accurate drilling signal.

Key words: Drilling signal; Winform; filter processing; mixed programming; dynamic link library call

**目录**

[1 绪论 1](#_Toc9241)

[1.1 研究背景与意义 1](#_Toc25082)

[1.1.1 研究背景 1](#_Toc22409)

[1.1.2 研究意义 1](#_Toc11900)

[1.2 滤波研究历史及发展 2](#_Toc29486)

[1.2.1 国外滤波研究历史 2](#_Toc14116)

[1.2.2 国内滤波研究进展 2](#_Toc10511)

[1.3 本文研究的主要内容 3](#_Toc32284)

[2 滤波理论与开发平台概述 5](#_Toc4601)

[2.1 滤波原理与频率特性 5](#_Toc8963)

[2.2 滤波概述 8](#_Toc14583)

[2.2.1 模拟滤波 8](#_Toc20241)

[2.2.2 数字滤波 8](#_Toc11088)

[2.3 Winform特点与介绍 10](#_Toc1952)

[3 带通滤波器设计方案 12](#_Toc31094)

[3.1 带通滤波器设计路线 12](#_Toc16850)

[3.2 数字带通滤波器设计方法 13](#_Toc14491)

[3.2.1 窗函数法 14](#_Toc10329)

[3.2.2 频率样本采样法 14](#_Toc20933)

[4 数据处理与滤波平台的实现 16](#_Toc19644)

[4.1 现场数据分析 16](#_Toc4755)

[4.2 滤波器及界面的实现 21](#_Toc1934)

[4.2.1 Hanming滤波实现 21](#_Toc21775)

[4.2.2 优化滤波实现 24](#_Toc10084)

[4.3 界面设计与结果分析 26](#_Toc32585)

[4.3.1 界面显示效果 26](#_Toc19612)

[4.3.2 结果分析 30](#_Toc24717)

[5 结论与展望 32](#_Toc44)

[5.1 结论 32](#_Toc24938)

[5.2 展望 32](#_Toc17935)

[致谢 33](#_Toc25840)

[参考文献 34](#_Toc12308)

# 1 绪论

## 1.1 研究背景与意义

### 1.1.1 研究背景

随着石油开采工业的飞速进步与开展，石油勘探开发的重点逐步转向深井、超深井。井下钻具振动是现阶段钻井工程中广泛存在的情况，剧烈的井下振动会加剧钻具的失效，缩短钻具的使用寿命，是导致钻具疲劳的主要原因，而且可能会形成更大的经济损失。对于钻具振动情况的掌握必须通过对钻具信号的采集获得，所以获得准确的钻具信号是至关重要的，而要获得准确的钻具信号就必须进行滤波处理，因此对钻具信号实行必要的滤波以解决相关问题是必要的。很多研究证明，钻过程中钻具的振动频率受很多因素的干扰，包括钻具自身特性、钻具动力源、岩壁与钻具的撞击。因此，对钻具信号进行滤波处理，在钻具信号处理中是十分重要的。钻具振动信息影响着对钻具运行环境与状态的判断，精确的提取钻具信号关系着整个钻井信号处理的过程。

研究表明，钻具在井底的整个运行环境非常复杂，其中包括含有钻井液的井眼，钻井液及地底其他液体的压力等，运行情况和受力条件非常复杂。钻井过程中钻具钻进的方式是旋转式和滑进式，所以在工作时往往伴随着各种各样的振动。而这些振动的信号中含有丰富的干扰信号，因此对钻具信号进行滤波处理是信号处理中尤为重要的环节。

### 1.1.2 研究意义

伴随着各种问题的提出，很多学者对此作出了大量研究分析。在信号的滤波处理方面，国外主要的方法以多抽样率数字信号处理、自适应数字滤波器、谱估计及其分析等现代数字信号处理理论为代表，同时结合DSP、DFT、FFT等工具，技术相对较为成熟。在国内有数学形态学的方法来处理现场采集的旋转机械振动信号的方法等，但国内该系列的钻研主要是通过研究理想的模型和进行理想论述的计算得到与实践运用有很大差异。本论文介绍采用应用于工程实际的Winform平台设计一种可在工程实际中运行的钻具振动滤波器，对于钻具信号滤波处理在现场工程应用具有重要的意义，同时对于其他现场工程的振动信号处理也具有重大意义。

## 1.2 滤波研究历史及发展

### 1.2.1 国外滤波研究历史

在钻井过程中，钻具振动通常与钻具及其组成结构的动力学特征及性质相关。在研究钻具振动理论同时人们已经开始了振动信号滤波方法的研究。国外发展起来的滤波处理方法主要有以下几种典型代表：

Koskinen 等人提出了建立在集论基础上的，几何形态学分析和描述相结合的工具，即软数学法。

研究发现，信号权值与成份形成元素相关，并由中心和软边界线两部分组成。它与硬数学形态学有着十分相似的代数特性，但后者比前者具备更强的抗噪声与抗干扰能力，更好的滤波处理效果。

Gasterato将模糊理论总集理论实践到软数学形态学提出了模糊软数学形态学。

该措施能够依据图像的拓扑构造，正确选择模糊总集运算操作数及结构元素核心、软界限的定义域，并经过改动反映结构元素与图像间相似程度的数 K的值调整相关信号滤波效果图像处理的数据输出结果。

Goustias和Heijmans提出形态小波理论。

两种滤波方法是将形态学和小波变换相互结合，提出来的一种全新的非线性、多分辨信号滤波方法。形态学滤波法是以数学形态学为基本，同时基于解决信号的数学几何结构特征，并具有并行快速滤波实现的特点。它使应用之前定义出来的相关构造元素对处理信号进行相互匹配或局部修正，以达到提取理想信号，抑制噪声的目的。小波变换则是用伸缩与平移小波时形成的小波基来改变（分解）或反改变（重构）时变信号的进程。将两者有机的结合，形成的形态小波滤波论述时常受到了国内外学者的广泛钻研和密切关注。

Widrow.B于1967年提出了自适应滤波方法。

自适应滤波的一般原理就是频率自主适应数据输入信号而变化的，其适用范围更广。在没有知晓任何信号和噪声的相关信息的条件下，该滤波措施则可应用前一时间已取得的滤波参数来主动调整目前时刻的滤波参数，以完成滤除信号和噪声未知或随机变换的统计特征，从而达到最优滤波处理结果。

### 1.2.2 国内滤波研究进展

在对信号处理的研究过程中，我国学者在信号滤波、降噪方面也取得了不小的成绩。

胡爱军提出了一种基于数学形态学的处理措施来对旋转机械振动信号进行滤波。应用这种方法可以有效地消除噪声，消除干扰。在针对不同类型、不同强度噪声干扰的情况下，设计了开－闭和闭－开组合的数学形态滤波器，通过仿真结果与实例实验结果显示，该种类型的形态滤波器滤波特性与性能有非常好的效果。同时也表明该类滤波器能有效地剔除具有干扰特征的脉冲噪声和随机噪声，并且滤波处理的效果明显，且算法的实现相对比较简单。

杨全进等人研究的无迹卡尔曼滤波处理措施。无迹卡尔曼滤波处理是采用一组代表着均值和方差的分布的采样点来对非线性系统采样非线性方法进行计算，在不对非线性方程线性近似的条件下，实现线性卡尔曼滤波器的滤波功能。

周琪等人研讨的基于陀螺传感器的姿态解算滤波处理措施。这种滤波算法基于姿态角微分方程建立系统的状态方程模型，利用互补滤波后的姿态角作为系统的观测量，再应用扩展到基于陀螺仪传感器的采集数据与加速度计、电子罗盘的测量数据，进行综合分析，从而达到对信号滤波处理的效果。

张建成等人设计了一种改良的 EMD和形态滤波相结合的齿轮系统故障特征提取新处理措施。

我国的许多学者对钻具信号滤波问题进行了非常多的研究，尽管取得了很多成绩，但该系列的研究主要是靠建立理想化的模型和数学计算得到的与实践运用有很大差异。

## 1.3 **本文研究的主要内容**

针对国内对钻具信号滤波的发展，在以往的情况中并没有进行能够在工程实际中进行滤波处理的研究，为了实现能够在现场工程中对钻具信号进行滤波处理，本论文的研究设计一种基于Winform的带通滤波器，可在现场工程中进行滤波处理。钻具粘滑振动信号可以通过测量下动态转速与地面转速建立识别模型来识别粘滑现象，下转速可以通过地面录参数扭矩来估算，因为扭矩不是严格的周期性信号，而是通常具有很宽的频率范围，所以精确的计算是非常复杂的。完成整个基于Winform的钻具信号带通滤波器设计就必须从以下主要内容进行研究。

（1）本次设计主要研究内容如下：

1. 采用Visual Studio软件，设计Winform窗体程序实现非周期信号的处理系统。
2. 分析钻具信号的特性选取合适的滤波器完成对信号的带通滤波处理。分析滤 波器的特性，对比不同滤波器对不同采集信号特性的影响。
3. 完成采集录井参数的分析，根据录井参数的特点优选滤波器算法。
4. 设计Winform窗体程序及信号处理系统实现计算参数的带通滤波处理。
5. 完成滤波算法优化及带通滤波信号处理系统的优化。

（2）设计主要实现的功能

1. 滤波器的设计。
2. 绘制幅频曲线。
3. 计算振动系数。
4. Winform对滤波效果进行显示。

针对以上功能实现用户交互界面。

# 2 滤波理论与开发平台概述

## 2.1 滤波原理与频率特性

滤波器是一种选频装置，可以使信号中特定的频率成分在允许的范围内被得到，而极大地衰减允许范围外的频率成分。在测试滤波的过程中，运用滤波器的这种选频作用，能够滤除杂波噪声或进行频谱处理分析。从大的范围上来讲，任何一种信息传输的通道（媒质）都可视为是一种滤波器。由于任何滤波器的响应特性都是激励频率的函数，都可用频域函数说明其传输特性。所以，形成滤波测试系统的任何一个步骤，诸如机械系统、电气网络、仪器仪表甚至导线等等，都将在某个确定的频率范围内，按其频域特性，对所通过的信号采取变换与处理。

滤波器的构造结构是抑制干扰信号的有效措施，通常的滤波方式分为模拟滤波和数字滤波，传统的仪表设计一般采用模拟滤波，而数字滤波因其设计和实现上的灵活性、高精度、高可靠性等优势在智能仪表的设计中得到了广泛的应用。同时数字滤波作为数字信号采集与分析处理技术的重要组成，能满足滤波器对幅度和相位特征性的十分严格的要求，克服模拟滤波器所无法处理的电压和温度漂移以及噪声等情况。

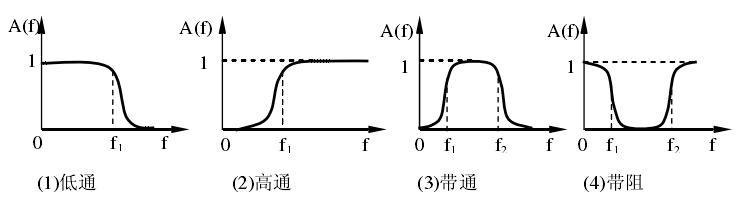


图2.1 滤波器的幅频特性

（1）滤波器的选频特征与原理

通常采用的滤波器有四种，分别是低通、高通、带通和带阻滤波器(如图2.1所示)。图中(1)是低通滤波器，在频段0-之间，幅频特征曲线接近平直，可以使信号中低于的频率成份能够不受衰减地通过，高于的频率成份则被极大的衰减使其基本不能通过；图中(2)的高通滤波器与低通滤波器的滤波效果刚好通带反向，在频段-∞幅频特性曲线是平直的，确保信号中高于的频率成份几乎不受衰减地通过，低于的频率成份受到极大地衰减；图中(3)是带通滤波器，其通过频带在-之间，信号中高于而低于的频率成份能够不受衰减地通过，其它频率成份则被抑制；图中(4)为带阻滤波器，其特性性质与带通滤波器相反。

而对于实际设计中往往会存在很多干扰因素，所以现实设计中所能使用的滤波器的频率特性与理想的情况下滤波器的频率特性之间存在着一些明显的差别，以低通滤波器的频率特性举例说明。

（2）理想滤波器的特性

假设滤波系统的原始信号为，信号中混入干扰，它们有不一样的频率成分。滤波器的单位脉冲响应为。则理想设计出来的滤波器的输出为：

 (2.1)

即干扰信号被滤除，而信号无失真只需延时和线性增加。对式(2.1)作傅里叶变换得：

 (2.2)

假设干扰信号被滤除，即 ：

 (2.3)

将代入式整理得：

 (2.4)

假定信号频率成分为：，噪音频率成分为。则完成滤波的理想低通滤波器特性是：

 (2.5)

即：  (2.6)

 (2.7)

系统的单位脉冲响应为：

 (2.8)

理想情况下低通滤波器的频率特征曲线如图2.2所示，单位时间内的脉冲的波形如图2.3所示。

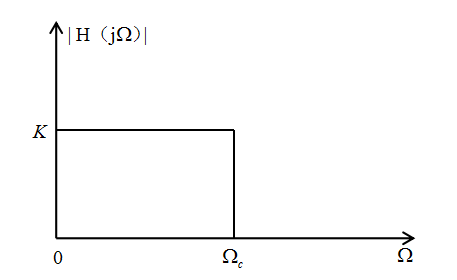


图2.2 理想低通滤波器频率特性

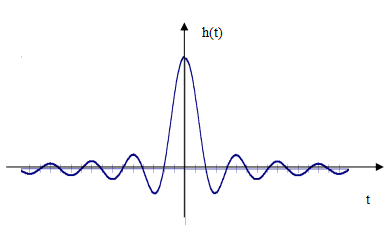


图2.3 理性滤波器的单位脉冲响应()

理想情况下的滤波器的效果有无限长、非因果的单位脉冲响应和间断的频率特性，要用十分平稳的线性时不变系统来完成这样的特征，在实际情况中是不可能的。工程上所使用是用脉冲响应为具备不间断频率特征的线性时不变系统或有限长的、有因果关系的、相对确定的线性时不变系统来拟合接近理想特性。在满足要求范围内的误差的要求的前提下实现理想滤波器的特征。所以实际的滤波器的频率特征曲线如图2.4所示。



通 带

阻带

过渡带

图2.4 实际滤波器的频率特性



其中：

——截止频率

——阻带起始频率

——过渡带宽

在通带内幅度响应以的误差接近于1，即：

 (2.9)

为阻带起始频率，在阻带内幅度响应以小于的误差接近于零，即：

 (2.10)

为了使逼近理想低通滤波器的方法成为可能，还必须提供一个带宽为的不为零的过渡带。在这个频带内，幅度响应从通带平滑的下落到阻带。

## 2.2 滤波概述

### 2.2.1 模拟滤波

电路实现的滤波器称为模拟滤波器分为无源滤波器和有源滤波器。无源滤波器又称LRC滤波器，是利用电感、电容和电阻的组合设计构成的滤波电路，可滤除某一次或多次谐波。常用的类型有单调谐滤波器、双调谐滤波器、高通滤波器。有源滤波器是用电流互感器采集直流线路上的电流，经A/D采样，将所得的电流信号采用谐波离散算法来处理，得出谐波参考信号，作为PWM的调制信号，与三角波比较，从而得到开关信号，用此开关信号去调节IGBT单相桥，依据PWM技术的原理，将上下桥臂的开关信号反接，就可得到与线上谐波信号大小相等、运动相反的谐波电流，将线上的谐波电流抵消掉。这是前馈调节步骤。再将有源滤波器接入点后的线上电流的谐波参量反馈回来，作为系统调节器的输入信号，调整前馈调节控制的误差。

### 2.2.2 数字滤波

数字滤波器是用数字计算机对数字信号采取处理，处理方法就是依照提前编写好的计算机程序进行计算，用于完成信号滤波处理功能。用只有一定范围的精度算法完成的间断时间间隔线性非时变系统，其待处理的信号是一组数字量，其得到的结果是通过滤波后的另一组数字量。所以，它本身即可以是用数字硬件装配成的一台实现给定运算的专业数字计算机，也可以是将所需运算编成计算机语言，让通用计算机来执行。

（1）FIR数字滤波器

FIR又称为有限范围的单位冲激激励响应滤波器，指在一定的信号范围内系统的单位脉冲激励响应仅有非零值的滤波器，是数字信号处理系统中最基本的元件，它能够在保证任何幅频特性的同时具有严格的线性相频特征，同时其单位抽样激励响应是有一定范围的，因此滤波器是平稳的系统。为了使信号滤波效果能够不发生失真，信号的采样速度一定要满足奈奎斯特定理，平时取信号频率的截止频率的1020倍做为采样频率，普遍可用速度较高的逐次逼进式A/D转换器。M阶FIR滤波器的系统函数为：

 (2.11)

其中是的M阶多相式，在一定范围内的平面内有M个零点，在平面上原点=0有M个极点。

FIR滤波器的频率响应为：

 (2.12)

它的另一种表示方法为：

 (2.13)

其中和分别为整个滤波系统的幅度激励和相位响应。

若系统的相位响应满足下面的条件

 (2.14)

即系统的群延时是一个与没有关系的常数，称为系统具备较好的线性相位。

如果一个离散系统的频率响应可以表示为：

 (2.15)

其中和是与无关联的常数，是可正可负的实函数，则称滤波效果是广义线性相位的。

如果M阶FIR滤波器的单位响应是实数，则可以证明系统是线性相位的充要条件为：

 (2.16)

当满足，称h[k]偶对称。当h[k]满足，称奇对称，按阶数又能够分为M为奇数和M为偶数，所以线性相位的FIR滤波器能够有四种类型，四种线性相位FIR滤波器的性质如附表1所示 。

（2）IIR数字滤波器

IIR又称为无限冲激激励数字滤波器。其转移函数的极点应位于面的单位圆内，以确保滤波器滤波结果的稳定性。这类数字滤波器完成滤波优劣为递归型构造，故又称一维递归型数字滤波器。IIR数字滤波器具备以下几个特点：

1. 系统的单位冲激响应无限长。
2. 结构上是递归型的。
3. 系统函数在有限平面上有极值点。

无限冲激响应数字滤波器的设计步骤过程最重要的是依据模拟滤波器设计理论，由模拟滤波器特征或组成导出数字滤波器转移函数或构造，较典型的有冲激响应不变法和双线性变换法。原因为：

1. 模拟滤波器的设计技术相对成熟，可以广泛利用。
2. 模拟滤波器有大量的参考程序和表格。
3. 它的解可以为闭合形式的。

## 2.3 Winform特点与介绍

Winform是Visaul Studio中的一款用于设计用户界面的平台，Visaul Studio中进行编程的语言为C#语言，是一种面向对象的、运行于.NET FIRamework之上的安全高端设计语言。并定于在微软职业开发者论坛上登台并为广大程序员采用。它包括了诸如单一继承、接口等语言。Visaul Studio中的编程语言是由C和C++衍生并能综合使用的面向对象的编程语言。同时综合了VB简易的可视化操作和C++的高编译效率，以其方便并强大的操作能力、优雅的语法格调、创新的语言编辑特性和便捷的面向组件编程的支持成为.NET开发的首选语言。所以Visaul Studio中的Winform平台有以下特点：

1. 性能强大：Windows窗体可用于设计窗体和可视控件，以建立丰富的基于Windows的运行程序。
2. 折叠控件灵便：Windows窗体提供了一套完整的控件，并且开发人员可以定义本人自身有特征的新的控件，另外Winform控件是指以输入或操作数据的对象。
3. 操作方便：数据提供控件管理平台提供易于衔接OLEDB和ODBC数据源的数据控件，包含SQL Server、Jet、DB2以及Orcle等。
4. 折叠向导确定：向用户提供建立窗体、数据处理、打包和部署等的离散向导。
5. 使用安全：Winform程序编写中继承了c#能够消除很多常见的C++编程错误，大大提高了平台的安全性。
6. 折叠数据管理：Windows窗体对数据库读取与导入能提供完整的步骤。能够读取数据库中的数据，并在窗体上显示和操作数据。

# 3 带通滤波器设计方案

## 3.1 带通滤波器设计路线

确定系统主要功能与模块，并充分调研分析国内外振动信号滤波分析技术的现状，结合现场实际情况，确立主要设计的技术方向，并对相关技术和模型进行文献调研。

根据文献资料与功能需求的调研，研究分析钻具信号的特性，提取现场数据库，分别采用两种滤波算法对现场数据进行计算，求出两种算法传递函数的参数以及其他相关参数，最后结合现场真实数据对算法进行编写，使其功能可实现用户界面化，从而形成一套完整的基于C#的钻具信号数字滤波处理的总体方案。

最后，根据系统总体设计方案，同时结合现场的数据完善优化滤波参数，设计出一套信号滤波方案，并基于SQL Server 2008数据库管理软件和Mcrosoft.Net FIRamework4.0框架，采用C#中的Winform实现平台设计开发，设计滤波平台界面。进行现场数据测试，检验软件系统的运行稳定性和可靠性。

本论文介绍的带通滤波器设计策略是主要以两种滤波算法进行滤波处理，分别是：Hanming窗函数法和优化滤波法。通过对现场数据的初步处理分析，结合两种滤波的算法结构和要求对现场数据进行深化的滤波处理，采用MATLAB与Winform进行混合编程，然后再对滤波结果进行分析与对比，优化滤波算法最终得到优化后的滤波效果，最终显示在Winform用户界面行。待处理信号的来源主要通过读去现场采集上来存储在execl中的数据，现场数据则是由钻井现场通过传感器采集处理得到，包括 A/D 转换等采样控制等电路模块。其中，A/D 采样控制模块用来实现现场采集的待处理模拟信号的模数转换及采样控制，最后将数据存入数据库，方便用户调取。滤波器实现采样数据的滤波处理以滤除不必要的频率成分，便于其他数据处理及分析。

（1）钻具信号带通滤波器设计

将带通滤波器的通带频率范围设计为为0.2Hz-2Hz，中心频率为1Hz，同时在这个范围内，滤波带宽越窄越能达到滤波要求。采样频率大约为2000Hz左右，采集系统的采样频率才能保证目标频带范围内信息的完整性，为了实现滤波的功能，可通过改变滤波算法参数实现。

（2）本文的研究思路及流程图



图3.1 研究思路及流程图

## 3.2 数字带通滤波器设计方法

数字滤波器有较为基本的滤波器，也有窗函数法、频率样本采样法，基本的滤波器是直接截取通频带所要求的频率段，然后将其余频段的频率置为零，这样就实现了对频率的选频效果，也就完成了滤波的功能，而这种滤波方式会使所得到的频率依然出现很多干扰信号或者噪声信号，所以一般只适用于小数据滤波。对于大数据滤波我们通常采用的是窗函数法、频率采样法以及优选法等滤波方法，这些滤波方法能够在滤波中设置一个通带截止频率，一个阻带截止频率，另外这几类滤波方法更适用于窄带滤波。对于基本滤波器设计我们一般都会以以下步骤进行：

第一步，确定设计要求：在设计一个滤波器之前，必须有一些要求。这些要求要根据应用确定。

第二步，模型拟合：一旦确定了技术要求，就可运用之前学习过的根本原理和关系式，提出一个滤波器模型来拟合给定的指标体系。只是滤波器设计所要研究的主要问题。

第三步，设计完成：实行以上的两步之后获得的滤波器，通常是以差分方程，系统函数或脉冲激励来描述的。根据这个说明，可以用硬件或计算机软件来实现它。

### 3.2.1 窗函数法

窗函数法也称为傅里叶级数法，它的设计原理是从时域出发的一种滤波设计方法，它从理想频率特性出发，用该特性的傅里叶反变换作为滤波器系数，实现并改善数据的特性而加窗截断。这种的办法的根本思维是让待设计的实际滤波器拟合逼近理想特征。经过调整窗函数长度N能够有效地调节过渡带的宽度，缩小带内波动以及加大阻带的衰减。设计步骤如下：

第一步，根据对过度带宽及组带衰减要求，选择窗函数的类型并计算窗口长度N，窗函数类型可根据其组带最小衰减的条件独立选择，因为其长度N对组带最小衰减没有影响，在确定窗函数类型后，就可根据过渡带宽小于给定指标的条件，确定所拟用的窗函数长度N。设待求滤波器的过渡带为，它近似与窗口长度N成反比。窗函数确定后其计算公式也确定了。

第二步，根据待求滤波器的理想频率的理想频率响应求取单位脉冲响应，由FFT反变换求出：

 (3.1)

第三步，计算滤波器的单位脉冲响应。它是理想脉冲响应和窗函数的乘积。

第四步，验算技术指标是否满足要求。

### 3.2.2 频率样本采样法

频率样本采样法的基本思想是使所设计的FIR数字滤波器的频率特性在一些频率点上有准确的值，相当于在其它频率出的特性有较好的逼近。在实际使用中，为了设计与理想更为逼近的FIR滤波，采样值要满足一定的约束条件。

有限冲激响应滤波器的基本结构是一个分节的延时线，把每一节的输出加权累积，得到滤波器的输出。滤波网络传输函数为：。

根据频域的采样定理，对给定的理想滤波器频率响应进行等间隔的采样：

 (3.2)

把当作待设计的FIR数字滤波器的频率特性的采样值即：

 (3.3)

可得有限序列：

 (3.3)

利用频域的N采样点的值，通过式(3.4)求出滤波器的系统函数和频率响应。

 (3.4)

 (3.5)

其中是一个内插函数，即：

 (3.6)

所以每个采样点上，频率响应严格与理想特性一致，在采样点之间，频率响应由各采样点的内插函数延伸叠加而形成，因而有一定的的逼近误差，误差大小与理想频率响应的曲线形状有关。理想特性平滑，则误差小，反之误差大，并且在理想频率响应的不联系点出会产生肩峰和波纹，N增加，则采样点变密，内插误差就减少。

# 4 数据处理与滤波平台的实现

## 4.1 现场数据分析

由于接收到的现场数据有2个的TQA数据，数据类型为幅值，TIME的间隔=0.2。另外钻具的振动TQA数据会达到几千甚至上万个数据，而且数据是随着钻柱的不断钻进采集出来的，要想对钻具的振动信号进行滤波就得知道不同深度的钻具振动的信号，从而对其进行滤波处理，因此对它进行采样的过程中需要对数据进行深度采样。当数据通过深度采样采集出来之后要对数据进行幅频转换，比如对其进行进行FFT变换，将信号变换由时域变换到频域，在频域的基础上可以更加方便的对现场数据进行滤波处理进行滤波处理[8]。在对数据做滤波处理的时候，Winform难以进行频谱分析处理与滤波处理，所以采用MATLAB与Winform混合编程对现场数据做相关的分析处理，然后导出dll与.h文件，这类文件在Winform中是可以直接调用的，然后观察分析处理后的结果。

首先将TQA数据读入MATLAB软件中，然后通过窗口显示出两组数据的幅值图分别如下：



图4.1 第一组数据原始数据



图4.2 第二组数据原始数据

两组原始数据显示，钻具在运动的过程中有一段时间是停止的，并且在钻具开始运动的时候，钻具的振幅会出现过大的振动，这样会对钻具造成很大损害，同时在分析的过程中也会出现较大的误差，因此在对钻具信号进行分析的时候不能对所有的数据进行分析处理，而是选取其中比较稳定的一部分进行处理，将钻具停转的一段时间的数据截断不要，选择数据浮动在可接受的范围内的数据进行处理，这样才会是最后的滤波处理效果能够被很清楚的对比出来，才能显示出不同滤波方法的滤波处理效果的优劣。在对现场数据进行处理后，为了对现场数据进行滤波分析处理，我们需要对数据进行FFT变换。有FFT原理有FFT变换的公式为：

 (4.1)

FFT变换也有另外一种形式：

 (4.2)

FFT的实质是内积，三角函数是完整的正交函数集，不一样频率的三角函数的之间的内积为0，只需频率相等的三角函数做内积时才不为0。

 (4.3)

和求内积的时间里，仅有中频率为的分量才会有内积的结果，其他分量的内积为0。能够理解为在上的投影，积分值是时间从负无穷到正无穷的积分，就是把信号每个时间点在的分量叠加起来，能够理解为在上的投影的叠加，叠加的结果就是频率为的分量，也就形成了频谱。从现场数据分析知道数据某些地方具有较大的浮动，不能作为滤波分析的原始数据，所以要对现场数据进行分段取在钻具稳定运行时并且具有代表性的数据作为滤波分析的原始数据，在此基础上对数据进行滤波处理，并且分析他的幅频特性。所以在此基础上，对现场数据做出了以下的FFT变换。

（1）第一组数据：

由第一组数据所有的现场数据分析知道，在数据中会出现为零的数据段，首先对整段数据进行滤波分析处理得出的效果图如下：



图4.3 滤波后图

从图中可以看出钻具信号滤波后被拉低输出数据的幅值，并且在钻具从停止到运动的瞬间会有跳变，这些干扰因素必须被剔除后才能进行滤波处理，所以在进行滤波分析时选取钻具运动较为稳定的时间段进行分析。将现场数据中的500s到2000s截取下来，如图4.4所示，在这段时间里钻具振动相对稳定，没有出现突变信号，也没有出现钻具停转的情况，可作为原始数据进行滤波处理分析。



图4.4 截取后500s到2000s数据段

对500s到2000s段之间的数据进行FFT变换之后的FFT模值如下图：



图4.5 FFT变换图

同样也将现场数据中的4000s到6000s截取下来，如下图所示，在这段时间里钻具也没有出现突变信号，没有出现钻具停转的情况，所以也可作为原始数据进行滤波处理分析。



图4.6 截取4000s到6000s数据段图

对4000s到6000s段之间的数据进行FFT变换之后的FFT模值图如下图：



图4.6 FFT模值图

（2）第二组数据：

对第二组数据数据的整段数据进行滤波分析，得出滤波效果图如下图:



图4.7 滤波后图

由第二组数据的现场数据整段滤波效果看出，滤波信号也被整体拉低，也有突变值的存在，这样进行滤波处理没法看出滤波设计是否达到要求，所以选取钻具信号中不为零的数据段即钻具运动稳定的时间段，数据才能够反映出第二组数据钻具稳定运行过程中的振动信号的真实情况反映。

将现场数据中的1s到10000s截取下来，如下图所示，在这段时间里钻具振动相对稳定，可作为原始数据进行滤波处理分析。



图4.8 截取数据段图

现场数据中的1s到10000s之间的数据进行FFT变换之后的FFT模值图如下：



图4.9 FFT模值图

将现场数据中的14500s到16253s截取下来，如图4.10所示，在这段时间里钻具振动相对稳定，也可作为原始数据进行滤波处理分析。



图4.10 截取后数据段图

现场数据中的14500s到16253s之间的数据进行FFT变换之后的FFT模值图如下：



图4.11 FFT模值图

## 4.2 滤波器及界面的实现

### 4.2.1 Hanming滤波实现

Hanming窗口滤波法通常被称为升余弦窗，能有效的降低旁瓣的高度升高主瓣高度，可提供大于50dB的衰减，旁瓣峰值幅度为41dB，过度带宽8，阻带最小衰减53dB，并且能提供较小的阶数。本次研究要求的滤波要求为滤波频率带宽为0.02Hz-2Hz之内，中心频率为1Hz，并且要求滤波帯宽越窄也好。为了方便进行滤波处理，取频带为0.8Hz-1.3Hz并选择Hanming窗口法进行滤波设计。由Hanming窗口函数法的参数及对现场数据的分析，得出N为每次滤波所需数据段的长度，同时N的长度随着对不同数据段分析而不同。由Hanming窗口参数可将通带截频设置为=0.8Hz，=1.3Hz，阻带截频=0.1Hz，=2Hz。对两组现场数据的不同数据的段采用Hanming窗口法进行滤波处理，在MATLAB上显示出滤波的频谱图。

利用Hanming窗口函数对第一组数据数据在500s到2000s之间的数据进行滤波处理。



图4.12 幅频特性图

滤波后波形效果图如下：



图4.13 滤波后图

利用Hanming窗口函数对第一组数据数据在4000s到6000s之间的数据进行滤波处理。



图4.14 幅频特性图

滤波后波形效果如下：



图4.15 滤波后图

用Hanming窗口函数对第二组数据数据在1s到10000s之间的数据进行滤波处理分析。



图4.15 幅频特性图

滤波后波形效果如下：



图4.17 滤波后图

用Hanming窗口函数对第二组数据数据在14000s到16253s之间的数据进行滤波处理分析。



图4.18 幅频特性图



图4.19 滤波后图

### 4.2.2 优化滤波实现

由于使用Hanming窗口对噪声和干扰的衰减不够，而且大数据滤波处理中只使用一种窗函数滤波处理很得到准确的钻具信号，所以在对数据进行窗函数滤波的基础上另外结合频率样本采样法，从而使滤波效果更为平滑，钻具信号更接近真实信号，使得出的滤波效果更佳，所以在500s到2000s之间的数据采用在进行Hanming滤波法后保持滤波参数不变在对其进行优化滤波处理幅频特性如下：



图4.20 幅频特性图



图4.21 滤波后效果图

用优化滤波对第一组数据在4000s到6000s之间的数据进行滤波处理。



图4.22 幅频特性图



图4.23 滤波后

用优化滤波法对第二组数据在1s到10000s之间的数据进行滤波处理。



图4.24 幅频特性图



图4.25 滤波后效果图

用优化滤波法对第二组数据在14500s到16253s之间的数据进行滤波处理。



图4.26 幅频特性图



图4.27 滤波后效果图

结合以上两种滤波方法对钻具信号的频谱分析与滤波后效果图得出如果数据段中没有为零的数据，则在滤波后数据得出的结果为正常信号，所以在进行滤波时为了使滤波处理效果正常，必须采用一定的手段在滤波前进行数据处理，方便滤波处理能够正常有效的进行。

## 4.3 界面设计与结果分析

### 4.3.1 界面显示效果

带通滤波器界面设计，由于从现场得知钻具振动的频率范围在0.02Hz-2Hz内，本论文设计拟采用两种滤波方法进行对现场数据进行滤波处理，同时将信号通过的带宽设置为0.8Hz-1.3Hz，由于之前进行过了现场数据的分析处理，因此在Winform上只需有滤波器选项，一些基本参数（如：采样频率）以及衰减倍数即可，通过在Winform上进行对控件的使用，还有对chart控件的编程处理，整个滤波器的显示界面设计成如下图所示。界面设计程序见附录。

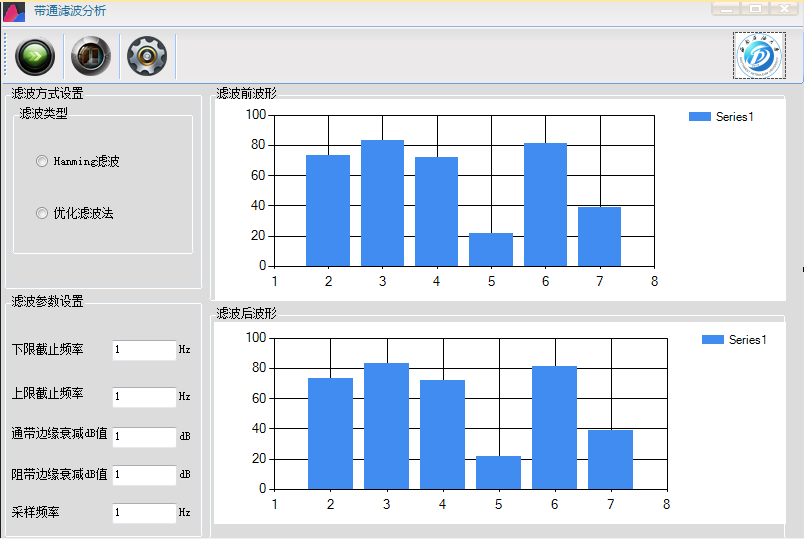


图4.28 带通滤波器显示界面

整个显示界面中有滤波开始与结束按钮，有滤波方式设置框架，可选择不同的滤波算法然后显示滤波图像，同时还有滤波参数设置框架。能够显示钻参数原始的TQA数据即在图中滤波前波形chart中显示，滤波后的数据图在第二个chart中显示，这样就完成了此次设计的带通滤波器的界面制作。在对滤波器进行操作之前先对其进行相关参数，由于滤波参数已经确定，可将参数设置如下图：

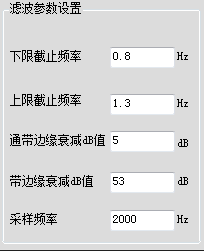


图4.29 参数设置板面

由于要求设计的滤波带宽过窄，通过对滤波算法的研究发现，滤波带宽越窄，需要设计的阶数也就相应较高，在完全依靠Winform的基础下没法对数据进行频谱分析和滤波处理，因为在Winform中直接使用C#语言进行滤波算法的编程在现实中很难达到要求的效果，所以本论文为了实现基于Winform设计滤波器，采用的方法是使用MATLAB和Winform混合编程进行滤波算法的研究，在MATLAB中编写自己计算出的FIR数字滤波算法，然后生成dll与.h文件，由于要使Winform的带通滤波器具有实际工程效果，所以我们在Winform中结合C#语言对MATLAB生成的dll与.h文件进行调用，然后在Winform数据处理中调用.h头文件进行滤波处理，最终完成滤波算法在Winform平台上的实现。以下是在Winform平台上滤波效果的显示图。

首先采用Hanming滤波法对第一组数据的TQA完整的现场组数据进行滤波处理，得出的滤波效果如下：

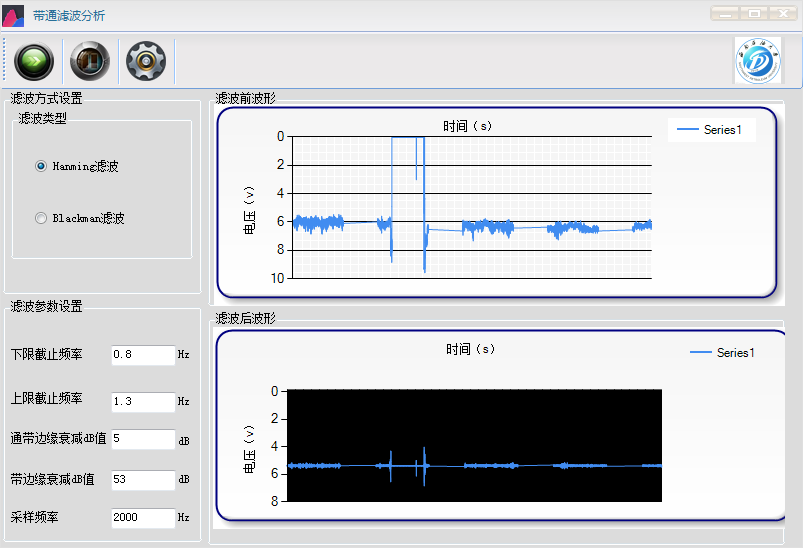


图4.30 Hanming滤波效果图（第一组数据）

然后采样最优滤波法对第一组数据的TQA完整的现场组数据进行滤波处理得出的滤波效果如下：

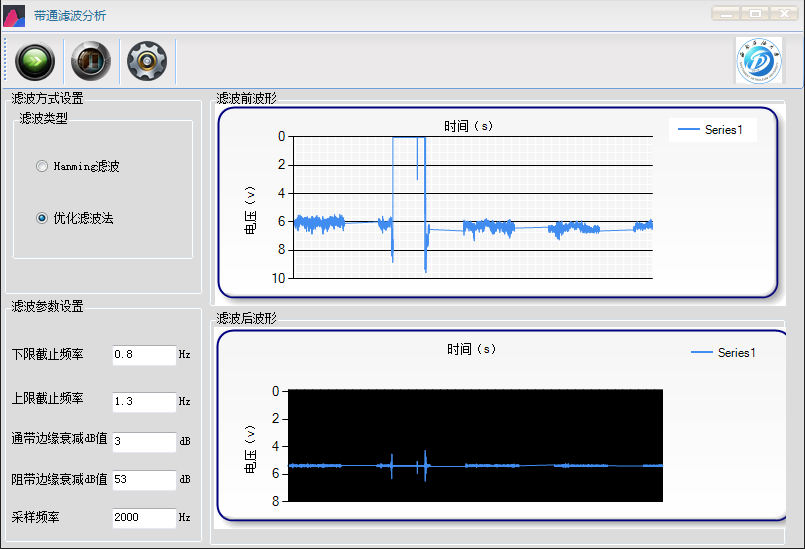


图4.31 优化滤波效果图（第一组数据）

选择优化滤波，由于在对整组滤波数据进行滤波处理后发现滤波后的效果虽然达到了但整体的滤波信号幅值被拉低了，所以在对第二组数据进行滤波时采用截断数据为零的数据段截取时间段在10000s之间的数据进行滤波，滤波参数的设置不变，进行滤波后在Winform上显示效果如下：

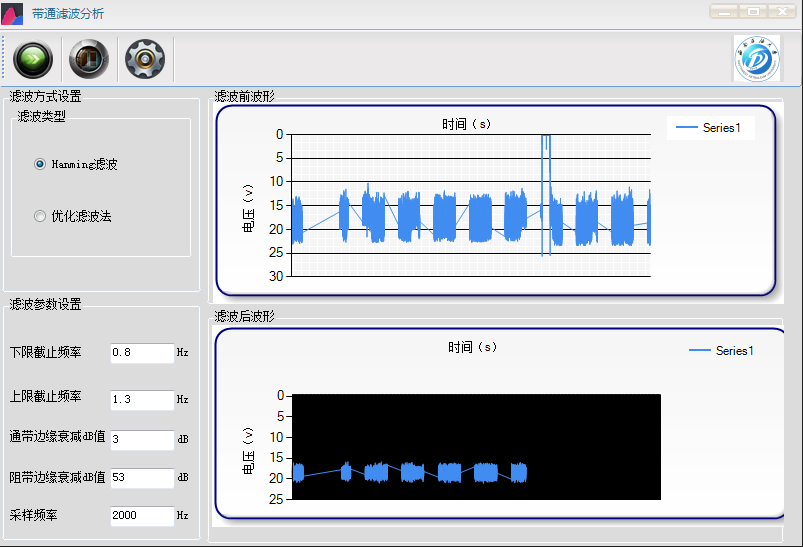


图4.32 Hanming滤波效果图（第二组数据）

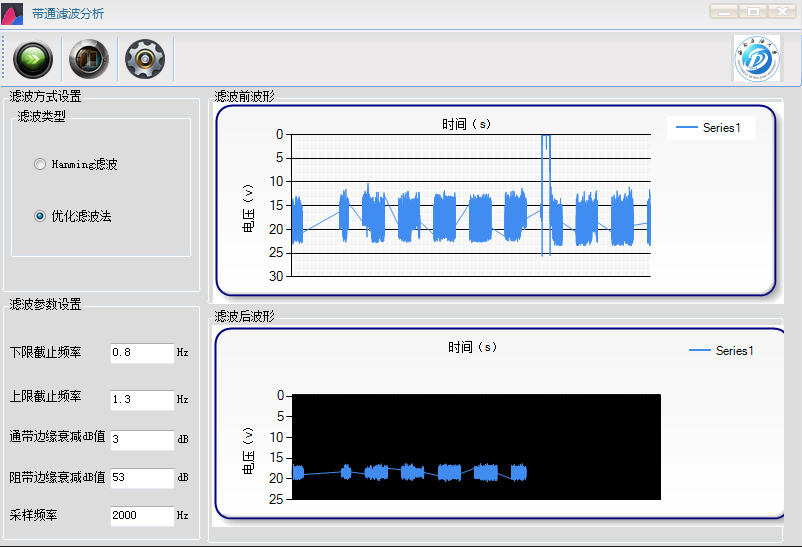


图4.33 优化滤波效果图（第二组数据）

### 4.3.2 结果分析

综合两种滤波方法的界面显示效果发现，当数据中出现为零的数据即钻具停止转动时，滤波后会出现一定幅值保持不变的信号，而当钻具从运行到停止和开始运行的时刻都会出现突变现象，而且在整段数据中如果出现为零的数据，那么在滤波后滤波数据的整体幅值会被拉低，这样严重影响了我们的滤波处理，因此，为了使滤波后的数据可用于对钻具运行中的钻具振动信号分析，应对现场数据中为零的数据进行剔除，这样得出的滤波数据才是真实的钻具振动滤波处理结果，滤波效果才更准确、更具有现场实用性。由两种滤波效果图的显示，在同一组数据中的数据用优化滤波法效果与平滑性要较好，更加具有采用性。

# 5 结论与展望

## 5.1 结论

本文设计了一种基于Winform平台显示的钻具振动带通滤波器，完成了任务书要求中的相关要求，主要完成了以下几点：

1. 应用C#完成Winform窗体主体程序设计；
2. 分析滤波器的特性，对比不同滤波器对不同采集信号特性的影响；
3. 完成采集录参数的分析，根据录参数的特点优选滤波器算法；
4. 设计Winform窗体程序及信号处理系统实现计算参数的带通滤波处理;

通过对本次毕业设计的制作，深入学习了滤波的原理与方法，也知道了钻具现场数据的一些分析方法，学会了很多课堂中没有学到的知识，同时也加强了自己的动手能力以及单独完成一个项目设计的能力，同时也学会采用不同的软件进行混合编程解决实际问题，虽然途中有很多不懂和迷惑，但在指导老师的帮助下解决了。由于钻具振动属于小频率波的振动，在对这类波的研究中我所接触到的知识稍显微薄，因此，在实现滤波的过程中我只采用了一次滤波，并且使用的是较为普通的窗函数法进行滤波处理，在滤波效果上能达到较为准确的得到钻具信号的要求，不足之处是没能将基于Winform的钻具振动带通滤波器开发设计成完整的应用平台，整个完成的程度没有完全达到预期期望的效果。

## 5.2 展望

对钻具信号的滤波在整个钻具信号的采集中是至关重要的一个环节，而现在很多对于钻具信号数据的滤波处理大多是将大数据采集后送往实验室进行滤波处理后然后才进行钻具情况的判定，这样很大情况下限制了评判的实时性，同时也不能及时预防钻工程的安全事故，本文研究的基于Winform的钻具信号带通滤波器能够将滤波过程直接在现场进行，大大的减小了处理时间，提高了预防效果。基于C#强大的功能和能够应用于工程实际的优点，以及钻具信号数据获取需要非常精确的重要性，表明本文研究的基于Winform钻具振动带通滤波器在工程实际中有良好的发展前景，能够很好的对以后的石油钻井安全起很大的预防效果。在本论文虽然没有采用较好的滤波算法，也没有进行多次滤波处理，而在以后的继续开发与研究的过程中，可以将本文研究的方向开发成一套完整的滤波平台。这样不仅可以用于石油钻井行业，同时还可以应用于其他行业，比如：旋转机架振动信号采集、无人飞机振动信号采集等这些振动会对机械及人危害生命财产安全造成损害的行业。

# 致谢

历时一学期，经过文献的查阅与软件编程的学习，运用自己大学四年所学到的知识，最终完成本次毕业设计的研究。在整个设计的过程中我的指导老师郭老师，感谢他在百忙之中抽出时间对我进行一些技术上的指导，给我提供了很大的帮助，他教导我如何查阅知网等权威机构的资料，如何查阅外文文献，在论文的撰写过程中郭老师给了我能一些论文撰写的技巧，帮我奖惩论文的总体结构。另外郭老师每周会给我开一次例会，总结我们上一周的进展，这样很大程度上督促了我的设计进程。另外我还得感谢本专业的研究生师兄，在很多细节的问题上他们给我提供了一些切实可行的解决方法，帮助我学会怎么去运用已有的知识解决问题，同时还教了我很多以前不知道学习方法。最后我还得感谢我的同学、室友在完成毕业设计的过程中帮助我在一些小问题上找到解决办法，并且在我没法进行时给我鼓励，让我有动力继续努力学习，继续进行本次毕业设计和本论文的撰写，最终完成本论文。

# 参考文献

[1] 程佩青. 数字信号处理教程[M]. 北京: 清华大学出版社: 第二版, 2001.

[2] 梁宏伟. 钻具信号的研究与应用[D]. 北京: 中国石油大学, 2008.

[3] Kyllingstad A, Nessjøen P J. A new stick-slip prevention system[C]//SPE/IADCDrilling Conference and Exhibition. Society of Petroleum Engineers, 2009.

[4] 王小科. 学通C#的24堂课[M], 北京: 清华大学出版社，2011.

[5] Mble,Desplns,J.P.,Pvone,D.:mplementton of Shocknd VbrtonMtgtonProcess:New

W of Usng Surfce Meusurements to Detect Down Hole Vbrton,pperDC/SPE 36883 presented t,1996 DC/SPE Drllng Conference held nMlntl, 22-24 October 1996.

[6] 高岩, 陈亚西, 郭学增. 钻柱振动信号采集系统及谱分析[J]. 录井技术, 1998, 9(3): 44-51.

[7] 杨福生. 随机信号分析[M]. 北京: 清华大学出版社, 1990: 151-164.

[8] 顾福年, 胡光锐. 《数据信号处理》习题解答[M]. 北京：科学出版社，1983.

[9] 陈怀琛. 数字信号处理教程: MATLAB 释义与实现[M]. 电子工业出版社, 2004.

[10] 陈添. 钻具信号的采集及应用研究[D], 北京: 中国石油大学, 2003.

[11] 郑勇. 无人机飞行异常振动信号采集方法研究[J]. 科技通报, 2014, 30(7): 184-187.

[12] 宗孔德, 胡广书. 数据信号处理[M]. 北京: 清华大学出版社, 1988：12-26.

[13] Chen D C K, Comeaux B C, Gillespie G M, et al. Real-time downhole torsional vibration monitor for improving tool performance and bit design[C]//IADC/SPE Drilling Conference. Society of Petroleum Engineers, 2006.

[14] 高岩, 陈亚西, 郭学增. 钻柱振动信号采集系统及谱分析[J]. 录井技术, 1998, 9(3): 44-51.

[15] Tsuria Y, Handelman D. Theft prevention system and method: U.S. Patent 5,939,975[P]. 1999-8-17.

[16] 刘菲菲, 李建平, 陈国强. 坐姿下人体振动特性建模及实验分析[J]. 山东大学学报: 工学版, 2012, 42(4): 103-107.

[17] 汪颖, 刘奇, 李成鑫. 基于数学形态学的分水岭图像分割方法[J]. 四川大学学报: 自然科学版, 2004 (z1): 457-460.

[18] 刘铮. UKF 算法及其改进算法的研究[D]. 长沙: 中南大学, 2009.

[19] 刘斌. 滤波器的分类与应用[J]. 家电维修: 大众版, 2012 (9): 50-51.

[20] 梁宏伟. 钻具信号的研究与应用[D]. 中国石油大学硕士论文, 2015.

[21] 黄俊钦. 随机信号处理[M]. 北京：北京航空航天大学出版社, 1990.

[22] 刘本永. 非平稳信号分析导论[M]. 北京: 国防工业出版社, 2006.

[24] 王济, 胡晓. MATLAB在振动信号处理中的应用[M]. 北京: 中国水利水电出版社, 2006.

[25] 韩学岩, 武庆河. 钻具振动的三种基本形式[A]. 录井技术文集, 2004 (1): 194-199.

[26] 丛玉良, 王宏志. 数字信号处理原理及其MATLAB实现[M]. 北京：电子工业出版社, 2005.

[27] 中国电子仪器仪表学会信号处理学会, 振动数字信号处理程序库, 编委会. 振动数字信号处理程序库[M]. 北京: 科学出版社, 1988.

# 

**附录1：**

表1四种线性相位FIR滤波器的性质

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类型 | 一类 | 二类 | 三类 | 四类 |
| 阶数 | 偶数 | 奇数 | 偶数 | 奇数 |
| h[k]的对称性 | 偶对称 | 偶对称 | 奇对称 | 奇对称 |
| 关于=0的对称性 | 偶对称 | 偶对称 | 奇对称 | 奇对称 |
| 关于=的对称性 | 偶对称 | 奇对称 | 奇对称 | 偶对称 |
| 的周期 | 2 | 4 | 2 | 4 |

**附录2：**

窗体设计程序

usng Sstem;

usng Sstem.Collectons.Generc;

usng Sstem.Lnq;

usng Sstem.Text;

usng Sstem.Drwng;

usng Sstem.Drwng.Drwng2D;

usng Sstem.Dt;

nmespce UControl

{

publc clss chart

{

/// <summr>

/// 画曲线图

/// </summr>

/// <prm nme="dsPrmeter"></prm>

/// <returns></returns>

publc clss Curve2D

{

prvte Grphcs objGrphcs; //Grphcs 类提供将对象绘制到显示设备的方法

prvte Btmp objBtmp; //位图对象

prvte flot fltWdth = 480; //图像宽度

prvte flot fltHeght = 248; //图像高度

prvte flot fltXSlce = 50; //X轴刻度宽度

prvte flot fltSlce = 50; //Y轴刻度宽度

prvte flot fltSlceVlue = 20; //Y轴刻度的数宽度

prvte flot fltSlceBegn = 0; //Y轴刻度开始值

prvte flot fltTenson = 0.5f;

prvte strng strTtle = "曲线图"; //标括题

prvte strng strXxsText = "月份份; //X轴说明文字

prvte strng strxsText = "万元"; //Y轴说明文字

prvte strng[] strsKes = new strng[] { "一月"， "二t月"， "三▂月"， "四月"， "五月"， "六ⅷ月"， "七月"， "八月"， "九月"， "十月"， "十一月"， "十二t月" }; //键

prvte flot[] fltsVlues = new flot[] { 20.0f， 30.0f， 50.0f， 55.4f， 21.6f， 12.8f， 99.5f， 36.4f， 78.2f， 56.4f， 45.8f， 66.5f， 99.5f， 36.4f， 78.2f， 56.4f， 45.8f， 66.5f， 20.0f， 30.0f， 50.0f， 55.4f， 21.6f， 12.8f }; //值

prvte Color clrBgColor = Color.Snow; //背景色

prvte Color clrTextColor = Color.Blck; //文字颜色

prvte Color clrBorderColor = Color.Blck; //整体边框颜色

prvte Color clrxsColor = Color.Blck; //轴线颜色

prvte Color clrxsTextColor = Color.Blck; //轴说明文字颜色

prvte Color clrSlceTextColor = Color.Blck; //刻度文字颜色

prvte Color clrSlceColor = Color.Blck; //刻度颜色

prvte Color[] clrsCurveColors = new Color[] { Color.Red， Color.Blue }; //曲线颜色

prvte flot fltXSpce = 100f; //图像左右距离边缘距离

prvte flot fltSpce = 100f; //图像上下距离边缘距离

prvte nt ntFontSze = 9; //字体大小号数

prvte flot fltXRottengle = 30f; //X轴文字旋转角度

prvte flot fltRottengle = 0f; //Y轴文字旋角度 prvte nt ntCurveSze = 2; //曲线线条大小

prvte nt ntFontSpce = 0; //ntFontSpce 是字体大小和距离调整出来的一个比括较适合的数字

#regon 公共属性

/// <summr>

/// 图像的宽度

/// </summr>

publc flot Wdth

{

set

{

f (vlue < 100)

{

fltWdth = 100;

}

else

{

fltWdth = vlue;

}

}

get

{

f (fltWdth <= 100)

{

return 100;

}

else

{

return fltWdth;

}

}

}

/// <summr>

/// 图像的高度

/// </summr>

publc flot Heght

{

set

{

f (vlue < 100)

{

fltHeght = 100;

}

else

{

fltHeght = vlue;

}

}

get

{

f (fltHeght <= 100)

{

return 100;

}

else

{

return fltHeght;

}

}

}

/// <summr>

/// X轴刻度宽度

/// </summr>

publc flot XSlce

{

set { fltXSlce = vlue; }

get { return fltXSlce; }

}

/// <summr>

/// Y轴刻度宽度

/// </summr>

publc flot Slce

{

set { fltSlce = vlue; }

get { return fltSlce; }

}

/// <summr>

/// Y轴刻度的数值宽度

/// </summr>

publc flot SlceVlue

{

set { fltSlceVlue = vlue; }

get { return fltSlceVlue; }

}

/// <summr>

/// Y轴刻度开始值

/// </summr>

publc flot SlceBegn

{

set { fltSlceBegn = vlue; }

get { return fltSlceBegn; }

}

/// <summr>

/// 张力系数

/// </summr>

publc flot Tenson

{

set

{

f (vlue < 0.0f && vlue > 1.0f)

{

fltTenson = 0.5f;

}

else

{

fltTenson = vlue;

}

}

get

{

return fltTenson;

}

}

/// <summr>

/// 标括题

/// </summr>

publc strng Ttle

{

set { strTtle = vlue; }

get { return strTtle; }

}

/// <summr>

/// X轴数据

/// </summr>

publc strng[] Kes

{

set { strsKes = vlue; }

get { return strsKes; }

}

/// <summr>

/// Y轴数据

/// </summr>

publc flot[] Vlues

{

set { fltsVlues = vlue; }

get { return fltsVlues; }

}

/// <summr>

/// 背景色

/// </summr>

publc Color BgColor

{

set { clrBgColor = vlue; }

get { return clrBgColor; }

}

/// <summr>

/// 文字颜色

/// </summr>

publc Color TextColor

{

set { clrTextColor = vlue; }

get { return clrTextColor; }

}

/// <summr>

/// 整体边框颜色

/// </summr>

publc Color BorderColor

{

set { clrBorderColor = vlue; }

get { return clrBorderColor; }

}

/// <summr>

/// 轴线颜色

/// </summr>

publc Color xsColor

{

set { clrxsColor = vlue; }

get { return clrxsColor; }

}

/// <summr>

/// X轴说明文字

/// </summr>

publc strng XxsText

{

set { strXxsText = vlue; }

get { return strXxsText; }

}

/// <summr>

/// Y轴说明文字

/// </summr>

publc strng xsText

{

set { strxsText = vlue; }

get { return strxsText; }

}

/// <summr>

/// X轴说明文字颜色

/// </summr>

publc Color xsTextColor

{

set { clrxsTextColor = vlue; }

get { return clrxsTextColor; }

}

/// <summr>

/// X刻度文字颜色

/// </summr>

publc Color SlceTextColor

{

set { clrSlceTextColor = vlue; }

get { return clrSlceTextColor; }

}

/// <summr>

/// 刻度颜色

/// </summr>

publc Color SlceColor

{

set { clrSlceColor = vlue; }

get { return clrSlceColor; }

}

/// <summr>

/// 曲线颜色

/// </summr>

publc Color[] CurveColors

{

set { clrsCurveColors = vlue; }

get { return clrsCurveColors; }

}

/// <summr>

/// X轴文字旋转角度

/// </summr>

publc flot XRottengle

{

get { return fltXRottengle; }

set { fltXRottengle = vlue; }

}

/// <summr>

/// Y轴文字旋转角度

/// </summr>

publc flot Rottengle

{

get { return fltRottengle; }

set { fltRottengle = vlue; }

}

/// <summr>

/// 图像左右距离边缘距离

/// </summr>

publc flot XSpce

{

get { return fltXSpce; }

set { fltXSpce = vlue; }

}

/// <summr>

/// 图像上下距离边缘距离

/// </summr>

publc flot Spce

{

get { return fltSpce; }

set { fltSpce = vlue; }

}

/// <summr>

/// 字体大小号数

/// </summr>

publc nt FontSze

{

get { return ntFontSze; }

set { ntFontSze = vlue; }

}

/// <summr>

/// 曲线线条大小

/// </summr>

publc nt CurveSze

{

get { return ntCurveSze; }

set { ntCurveSze = vlue; }

}

#endregon

/// <summr>

/// 自动根据参数调整图像大小

/// </summr>

publc vod Ft()

{

//计算字体距离

ntFontSpce = FontSze + 5;

//计算图像边距

flot fltSpce = Mth.Mn(Wdth / 6， Heght / 6);

XSpce = fltSpce;

Spce = fltSpce;

//计算X轴刻度宽度

XSlce = (Wdth 2 \* XSpce) / (Kes.Length 1);

//计算Y轴刻度宽度和轴刻度开始值

flot fltMnVlue = 0;

flot fltMxVlue = 0;

for (nt = 0; < Vlues.Length; ++)

{

f (Vlues[] < fltMnVlue)

{

fltMnVlue = Vlues[];

}

else f (Vlues[] > fltMxVlue)

{

fltMxVlue = Vlues[];

}

}

f (SlceBegn > fltMnVlue)

{

SlceBegn = fltMnVlue;

}

nt ntSlceCount = (nt)(fltMxVlue / SlceVlue);

f (fltMxVlue % SlceVlue != 0)

{

ntSlceCount++;

}

Slce = (Heght 2 \* Spce) / ntSlceCount;

}

/// <summr>

/// 生成图像并返回图像对象

/// </summr>

/// <returns></returns>

publc Btmp Cretemge()

{

ntlzeGrph();

nt ntKesCount = Kes.Length;

nt ntVluesCount = Vlues.Length;

f (ntVluesCount % ntKesCount == 0)

{

nt ntCurvesCount = ntVluesCount / ntKesCount;

for (nt = 0; < ntCurvesCount; ++)

{

flot[] fltCurrentVlues = new flot[ntKesCount];

for (nt j = 0; j < ntKesCount; j++)

{

fltCurrentVlues[j] = Vlues[ \* ntKesCount + j];

}

DrwContent(ref objGrphcs， fltCurrentVlues， clrsCurveColors[]);

}

}

else

{

objGrphcs.DrwStrng("发生错误Vlues的长度必须是Kes的整数倍"， new Font("宋体"， FontSze + 5)， new SoldBrush(TextColor)， new Pont((nt)XSpce， (nt)(Heght / 2)));

}

return objBtmp;

}

/// <summr>

/// 初始化和填充图像区域，画出边框，初始标括题

/// </summr>

prvte vod ntlzeGrph()

{

//根据给定的高度和宽度创建一个位图图像

objBtmp = new Btmp((nt)Wdth， (nt)Heght);

//从指定的 objBtmp 对象创建 objGrphcs 对象即在objBtmp对象中画图)

objGrphcs = Grphcs.FIRommge(objBtmp);

//根据给定颜色(LghtGr)填充图像的矩形区域(背景)

objGrphcs.DrwRectngle(new Pen(BorderColor， 1)， 0， 0， Wdth 1， Heght 1); //画边框

objGrphcs.FllRectngle(new SoldBrush(BgColor)， 1， 1， Wdth 2， Heght 2); //填充边框

//画X轴，注意图像的原始X轴和轴计算是以左上角为原点，向右和向下计算的

flot fltX1 = XSpce;

flot flt1 = Heght Spce;

flot fltX2 = Wdth XSpce + XSlce / 2;

flot flt2 = flt1;

objGrphcs.DrwLne(new Pen(new SoldBrush(xsColor)， 1)， fltX1， flt1， fltX2， flt2);

//画Y轴

fltX1 = XSpce;

flt1 = Heght Spce;

fltX2 = XSpce;

flt2 = Spce Slce / 2;

objGrphcs.DrwLne(new Pen(new SoldBrush(xsColor)， 1)， fltX1， flt1， fltX2， flt2);

//初始化轴线说明文字

SetxsText(ref objGrphcs);

//初始化X轴上的刻度和文字

SetXxs(ref objGrphcs);

//初始化Y轴上的刻度和文字

Setxs(ref objGrphcs);

//初始化标括题

CreteTtle(ref objGrphcs);

}

/// <summr>

/// 初始化Y轴线说明文字

/// </summr>

/// <prm nme="objGrphcs"></prm>

prvte vod SetxsText(ref Grphcs objGrphcs)

{

flot fltX = Wdth XSpce + XSlce / 2 (XxsText.Length 1) \* ntFontSpce;

flot flt = Heght Spce ntFontSpce;

objGrphcs.DrwStrng(XxsText， new Font("宋体"， FontSze)， new SoldBrush(xsTextColor)， fltX， flt);

fltX = XSpce + 5;

flt = Spce Slce / 2 ntFontSpce;

for (nt = 0; < xsText.Length; ++)

{

objGrphcs.DrwStrng(xsText[].ToStrng()， new Font("宋体"， FontSze)， new SoldBrush(xsTextColor)， fltX， flt);

flt += ntFontSpce; //字体上下距离

}

}

/// <summr>

/// 初始化X轴上的刻度和文字

/// </summr>

/// <prm nme="objGrphcs"></prm>

prvte vod SetXxs(ref Grphcs objGrphcs)

{

flot fltX1 = XSpce;

flot flt1 = Heght Spce;

flot fltX2 = XSpce;

flot flt2 = Heght Spce;

nt Count = 0;

nt SlceCount = 1;

flot Scle = 0;

flot Wdth = ((Wdth 2 \* XSpce) / XSlce) \* 50; //将要画刻度的长度分段，并乘以50，以10为单位画刻度线。

flot fltSlceHeght = XSlce / 10; //刻度线的高度

objGrphcs.TrnslteTrnsform(fltX1， flt1); //平移图像(原点)

objGrphcs.RotteTrnsform(XRottengle， MtrxOrder.Prepend); //旋转图像

objGrphcs.DrwStrng(Kes[0].ToStrng()， new Font("宋体"， FontSze)， new SoldBrush(SlceTextColor)， 0， 0);

objGrphcs.ResetTrnsform(); //重置图像

for (nt = 0; <= Wdth; += 10) //以10为单位

{

Scle = \* XSlce / 50;//即( / 10) \* (XSlce / 5)，将每个刻度分五部分画，但因皑为以10为单位，得除以10

f (Count == 5)

{

objGrphcs.DrwLne(new Pen(new SoldBrush(xsColor))， fltX1 + Scle， flt1 + fltSlceHeght \* 1.5f， fltX2 + Scle， flt2 fltSlceHeght \* 1.5f);

//画网格虚线

Pen penDshed = new Pen(new SoldBrush(xsColor));

penDshed.DshStle = DshStle.Dsh;

objGrphcs.DrwLne(penDshed， fltX1 + Scle， flt1， fltX2 + Scle， Spce Slce / 2);

//这里显示X轴刻度

f (SlceCount <= Kes.Length 1)

{

objGrphcs.TrnslteTrnsform(fltX1 + Scle， flt1);

objGrphcs.RotteTrnsform(XRottengle， MtrxOrder.Prepend);

objGrphcs.DrwStrng(Kes[SlceCount].ToStrng()， new Font("宋体"， FontSze)， new SoldBrush(SlceTextColor)， 0， 0);

objGrphcs.ResetTrnsform();

}

else

{

//超过范围，不画任何刻度文字

}

Count = 0;

SlceCount++;

f (fltX1 + Scle > Wdth XSpce)

{

brek;

}

}

else

{

objGrphcs.DrwLne(new Pen(new SoldBrush(SlceColor))， fltX1 + Scle， flt1 + fltSlceHeght， fltX2 + Scle， flt2 fltSlceHeght);

}

Count++;

}

}

/// <summr>

/// 初始化Y轴上的刻度和文字

/// </summr>

/// <prm nme="objGrphcs"></prm>

prvte vod Setxs(ref Grphcs objGrphcs)

{

flot fltX1 = XSpce;

flot flt1 = Heght Spce;

flot fltX2 = XSpce;

flot flt2 = Heght Spce;

nt Count = 0;

flot Scle = 0;

nt SlceCount = 1;

flot Heght = ((Heght 2 \* Spce) / Slce) \* 50; //将要画刻度的长度分段，并乘以50，以10为单位画刻度线。

flot fltSlceWdth = Slce / 10; //刻度线的宽度

strng strSlceText = strng.Empt;

objGrphcs.TrnslteTrnsform(XSpce ntFontSpce \* SlceBegn.ToStrng().Length， Heght Spce); //平移图像(原点)

objGrphcs.RotteTrnsform(Rottengle， MtrxOrder.Prepend); //旋转图像

objGrphcs.DrwStrng(SlceBegn.ToStrng()， new Font("宋体"， FontSze)， new SoldBrush(SlceTextColor)， 0， 0);

objGrphcs.ResetTrnsform(); //重置图像

for (nt = 0; < Heght; += 10)

{

Scle = \* Slce / 50; //即( / 10) \* (Slce / 5)，将每个刻度分五部分画，但因为以10为单位，得除以10

f (Count == 5)

{

objGrphcs.DrwLne(new Pen(new SoldBrush(xsColor))， fltX1 fltSlceWdth \* 1.5f， flt1 Scle， fltX2 + fltSlceWdth \* 1.5f， flt2 Scle);

//画网格虚线

Pen penDshed = new Pen(new SoldBrush(xsColor));

penDshed.DshStle = DshStle.Dsh;

objGrphcs.DrwLne(penDshed， XSpce， flt1 Scle， Wdth XSpce + XSlce / 2， flt2 Scle);

//这里显示Y轴刻度

strSlceText = Convert.ToStrng(SlceVlue \* SlceCount + SlceBegn);

objGrphcs.TrnslteTrnsform(XSpce ntFontSze \* strSlceText.Length， flt1 Scle); //平移图像(原点)

objGrphcs.RotteTrnsform(Rottengle， MtrxOrder.Prepend); //旋转图像

objGrphcs.DrwStrng(strSlceText， new Font("宋体"， FontSze)， new SoldBrush(SlceTextColor)， 0， 0);

objGrphcs.ResetTrnsform(); //重置图像

Count = 0;

SlceCount++;

}

else

{

objGrphcs.DrwLne(new Pen(new SoldBrush(SlceColor))， fltX1 fltSlceWdth， flt1 Scle， fltX2 + fltSlceWdth， flt2 Scle);

}

Count++;

}

}

/// <summr>

/// 画曲线

/// </summr>

/// <prm nme="objGrphcs"></prm>

prvte vod DrwContent(ref Grphcs objGrphcs， flot[] fltCurrentVlues， Color clrCurrentColor)

{

Pen CurvePen = new Pen(clrCurrentColor， CurveSze);

PontF[] CurvePontF = new PontF[Kes.Length];

flot kes = 0;

flot vlues = 0;

for (nt = 0; < Kes.Length; ++)

{

kes = XSlce \* + XSpce;

vlues = (Heght Spce) + SlceBegn Slce \* (fltCurrentVlues[] / SlceVlue);

CurvePontF[] = new PontF(kes， vlues);

}

objGrphcs.DrwCurve(CurvePen， CurvePontF， Tenson);

}

/// <summr>

/// 初始化标括题

/// </summr>

/// <prm nme="objGrphcs"></prm>

prvte vod CreteTtle(ref Grphcs objGrphcs)

{

objGrphcs.DrwStrng(Ttle， new Font("宋体"， FontSze)， new SoldBrush(TextColor)， new Pont((nt)(Wdth XSpce) ntFontSze \* Ttle.Length， (nt)(Spce Slce / 2 ntFontSpce)));

}

}

}

}

**附录3：**

滤波程序

function ISD=Filter(S，N1，N2)

S=S(N1:N2);

N=length(S);

F1=0.1;F2=0.8;F3=1.3;F4=2;

dt=0.2;

df=1/(dt\*N);

f=df\*(0:N-1);

Sf=(fft(S));Temp=Sf(1);Sf(1)=0;

SD=HAMMING\_WINDOW(Sf，F1/df，F2/df，F3/df，F4/df);

plot(abs(Sf));

plot(S);

subplot(211)

plot(f，abs(SD));

xlabel('f(Hz)')

ylabel('幅值')

subplot(212)

SD(1)=Temp;

ISD=real(ifft(SD));

plot(0:N-1，ISD);

xlabel('t(s))')

ylabel('V(v)')

xlswrite('output.xls'，ISD);

end

function [Data]=HAMMING\_WINDOW(D，N1，N2，N3，N4)

N=length(D);

Data=D;

for i=1:N

if(i>=N1&&i<N2)

Hammingw=0.54+0.46\*cos(pi\*(i-N1)/(N2-N1)-pi);

Data(i)=Data(i)\*Hammingw;

elseif(i>N3&&(i<=N4))

Hammingw=0.54+0.46\*cos(pi\*(N3-i)/(N4-N3));

Data(i)=Data(i)\*Hammingw;

elseif(i<N1||i>N4)

Data(i)=0;

end

end

end

附录4：

滤波显示程序

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using Wavelet;

using Filter.h;

using System.IO;

using NTUIControl;

using System.Drawing.Drawing2D;

using System.Windows.Forms.DataVisualization.Charting;

using System.Data.OleDb;

namespace 带通滤波分析显示

{

public partial class frmWavelet : UForm

{

FileStream fs;

FileInfo fi;

StreamWriter sw;

private string filePath1 = "C:\\Users\\刘宁t\\Desktop\\结题\\新数据\\新数据Y1.xls";

private string filePath2 = "C:\\Users\\刘宁\\Desktop\\结题\\新数簓据\\新数据Y3.xls";

private DataTable dtData = new DataTable();

private bool isAxisAuto = true;

private DataTable dtPMudLog;

public frmWavelet()

{

InitializeComponent();

#region 曲线及面板样式

chart1.ChartAreas["ChartArea1"].AxisX.Title = "时间";

chart1.ChartAreas["ChartArea1"].AxisY.Title = "电压";

chart1.ChartAreas["ChartArea1"].AxisX.Interval = 1;

chart1.ChartAreas["ChartArea1"].AxisY.IsReversed = true;

chart1.ChartAreas["ChartArea1"].BackColor = Color.WhiteSmoke;

chart1.ChartAreas["ChartArea1"].BackHatchStyle = ChartHatchStyle.Cross;

chart1.ChartAreas["ChartArea1"].BackGradientStyle = GradientStyle.HorizontalCenter;

chart1.BackColor = Color.WhiteSmoke;

chart1.BackGradientStyle = GradientStyle.TopBottom;

chart1.BorderlineDashStyle = ChartDashStyle.Solid;

chart1.BorderlineColor = Color.Navy;

chart1.BorderlineWidth = 2;

chart1.BorderSkin.SkinStyle = BorderSkinStyle.Emboss;

chart2.ChartAreas["ChartArea1"].AxisX.Title = "时间";

chart2.ChartAreas["ChartArea1"].AxisY.Title = "电压";

chart2.ChartAreas["ChartArea1"].AxisX.Interval = 100;

chart2.ChartAreas["ChartArea1"].AxisY.IsReversed = true;

chart2.ChartAreas["ChartArea1"].BackColor = Color.WhiteSmoke;

chart2.ChartAreas["ChartArea1"].BackHatchStyle = ChartHatchStyle.Cross;

chart2.ChartAreas["ChartArea1"].BackGradientStyle = GradientStyle.TopBottom;

chart2.BackColor = Color.WhiteSmoke;

chart2.BackGradientStyle = GradientStyle.TopBottom;

chart2.BorderlineDashStyle = ChartDashStyle.Solid;

chart2.BorderlineColor = Color.Navy;

chart2.BorderlineWidth = 2;

chart2.BorderSkin.SkinStyle = BorderSkinStyle.Emboss;

#endregion

chart2.Legends[0].BackColor = Color.Transparent;

}

private void frmWavelet\_Load(object sender， EventArgs e)

{

string error = string.Empty;

string strConn = "Provider=Microsoft.Jet.OLEDB.4.0;" + "Data Source=" + filePath1 + ";" + "Extended Properties=Excel 8.0;";

OleDbConnection conn = new OleDbConnection(strConn);

string strExcel = "";

conn.Open();

OleDbDataAdapter myCommand = new OleDbDataAdapter(strExcel， strConn);

DataSet ds = new DataSet();

myCommand.Fill(ds， "table1");

conn.Close();

DataView dv = new DataView(ds.Tables[0]);

this.chart1.Series[0].ChartType = SeriesChartType.Line;

this.chart1.Series[0].Points.DataBindXY(dv，"TIME"，dv， "TQA");

}

private void rbUDP\_CheckedChanged(object sender， EventArgs e)

{

string ISD=string.Filter;

string error = string.Empty;

string strConn = "Provider=Microsoft.Jet.OLEDB.4.0;" + "Data Source=" +ISD + ";" + "Extended Properties=Excel 8.0;";

OleDbConnection conn = new OleDbConnection(strConn);

string strExcel = "";

conn.Open();

DataSet ds = new DataSet();

myCommand.Fill(ds， "table1");

conn.Close();

DataView dv = new DataView(ds.Tables[0]);

this.chart2.Series[0].ChartType = SeriesChartType.Line;

this.chart2.Series[0].Points.DataBindXY(dv， "TIME"，dv，"ISD");

}

private void tsbStart\_Click(object sender， EventArgs e)

{

if(rbUDP=true )

{

string ISD=string.Filter;

string error = string.Empty;

string strConn = "Provider=Microsoft.Jet.OLEDB.4.0;" + "Data Source=" +ISD + ";" + "Extended Properties=Excel 8.0;";

OleDbConnection conn = new OleDbConnection(strConn);

string strExcel = "";

conn.Open();

DataSet ds = new DataSet();

myCommand.Fill(ds， "table1");

conn.Close();

DataView dv = new DataView(ds.Tables[0]);

this.chart2.Series[0].ChartType = SeriesChartType.Line;

this.chart2.Series[0].Points.DataBindXY(dv， "TIME"，dv，"ISD");

}

if(rrbTCPC=true)

{

string ISD1=string.Filters1;

tring error = string.Empty;

string strConn = "Provider=Microsoft.Jet.OLEDB.4.0;" + "Data Source=" + ISD1 + ";" + "Extended Properties=Excel 8.0;";

OleDbConnection conn = new OleDbConnection(strConn);

string strExcel = "";

conn.Open();

OleDbDataAdapter myCommand = new OleDbDataAdapter(strExcel， strConn);

DataSet ds = new DataSet();

conn.Close();

DataView dv = new DataView(ds.Tables[0]);

this.chart2.Series[0].ChartType = SeriesChartType.Line;

this.chart2.Series[0].Points.DataBindXY(dv， "TIME"， dv， "ISD1");

}

}

private void rbTCPC\_CheckedChanged(object sender， EventArgs e)

{

string ISD1=string.Filters1;

tring error = string.Empty;

string strConn = "Provider=Microsoft.Jet.OLEDB.4.0;" + "Data Source=" + ISD1 + ";" + "Extended Properties=Excel 8.0;";

OleDbConnection conn = new OleDbConnection(strConn);

string strExcel = "";

conn.Open();

OleDbDataAdapter myCommand = new OleDbDataAdapter(strExcel， strConn);

DataSet ds = new DataSet();

conn.Close();

DataView dv = new DataView(ds.Tables[0]);

this.chart2.Series[0].ChartType = SeriesChartType.Line;

this.chart2.Series[0].Points.DataBindXY(dv， "TIME"， dv， "ISD1");

}

}

}