oof\_bk.py

这种线性函数的形式为: y

|= a \* x + b, 其中 y 是输出

值,x是输入值。

def func(x,p): func 函数实现了简单的线性 输出: | 函数,其中 a 和 b 是通过参 X,表示函数的自变量 P, 表示线性函数的系数 数向量 p 传递进来的。

返回线性函数计算得到 的结果, 即 a \* x + b

文件,该文件包含有关天文观测的配置数。具体而言,它包括了一些与观测标的信息:观测(SRC)、接收机(RCV) 频率(FREQ)、赤经(RAH)、 DECD)、校准(TCAL)、观测ā SER)、观测带宽(OBSW)、 LO)、以及最大 L 值(MA

fileName: 一个要读取 的 XML 文件路径的字

def \_\_init\_\_(self,fileName): 类包含一个构造函数 \_\_init\_\_, 该构造函数接收一个参数 |fileName,表示要读取的 XML 文 件的路径。构造函数将该路径存 储在类的属性 self.fileName 中。

obsConfigReader.py

class obsXmlReader:

def loadObsXml(self):

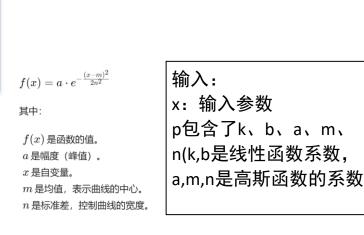
频率通道,这里会生成一个列表

self.bankEnd,表示每个通道的

self.fileName: 类的实

│打开 XML 文件,如果文件无法 │ │输出: 打开,则返回加载失败的信息。 | | 如果成功加载 XML 文件,返回一 │ 析 XML 文件内容。从 XML 文件 │ │其中 obs\_parameters 是包含加载 的根节点开始,提取其中的各种|的天文观测参数的元组。 天文观测参数,包括源名 如果加载失败,返回一个元组 (SRC)、接收机类型(RCV)、频率 (FREQ)等等。将提取的参数存储 在类的实例属性中,例如 self.SRC、self.RCV、self.obw 等 等。最后,将加载的参数封装成 一个元组,其中包含源名、接收 机类型、观测带宽、赤经、赤 纬、温度校准(TCAL)、观测者 (OBSER)、频率通道末端标识 (bankEnd)、频率通道对应的中 心频率(restFreq)、局放观测中心 频率(Lo)以及最大延迟(maxL)等 信息。如果 XML 文件中有多个





x: 输入参数

p包含了k、b、a、m

n(k,b是线性函数系数

a,m,n是高斯函数的系

z:实际观测到的数据

antF: 天线文件?

dtF:数据文件?

rcvTy:接收机类型。

Amp: 放大器增益

telF: 望远镜文件?

(可选参数,默认为字

multibeam: 是否是多

波束观测(可选参数,

Lo: 本振。

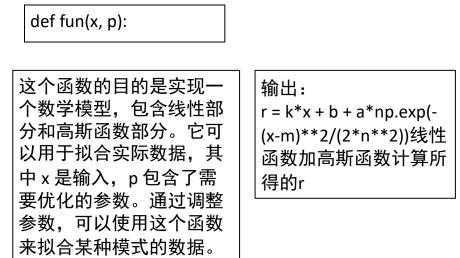
Bw: 带宽。

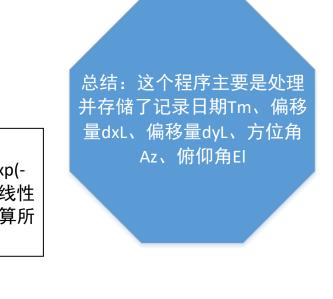
Sta: 站点。

Freq: 频率。

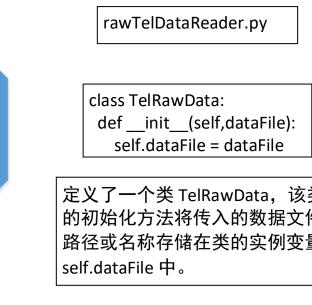
符串 "BB")。

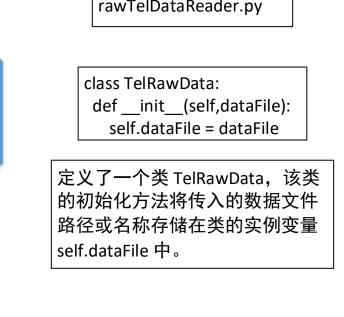
默认为 False)





self: 这是一个类方法,





(elevation)、偏移量

(dx、dy)、以及一个标

志(fg)。根据条件(Fg

是否等于 fg)筛选出符合

|中提取的年、月、日、 一时、分、秒使用slalib库计

|算将日期时间转换为%Y-

|%m-%d%H:%M:%S这种格

| 式,并将所有的数据存储

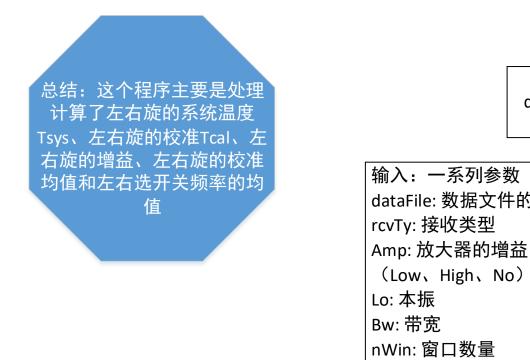
|到对应的列表中。最后,

将这些列表转换为 NumPy

数组,并返回这些数组的

|条件的数据。将从datatime

def getTelData(self,Fg): 打开指定路径的文件 (self.dataFile )存储了文 |记录日期(Tm)、偏 │所以第一个参数是类实││件路径)。初始化时间、 │ │移量X(dxL)、偏移量 方位角、俯仰角、x轴偏移 | Y (dyL)、方位角 和y轴偏移量的空列表用于 │存储文件提取的数据。然 | 后遍历文件的每一行。对 每一行进行处理,将其拆 分成数据列表。提取数据 列表中的各个数据包括日 期时间、方位角 (azimuth)、俯仰角



#### rawBkDataReader.py class DIBASRawData def init (self,dataFile,rcvTy,Amp,Lo,Bw,nWin,\ maxTm,minTm):

列属性,这些属性代表了| rcvTy: 接收类型 不同的数据信息。类的实 Amp: 放大器的增益 例化需要提供一些参数, (Low、High、No) |这些参数在初始化方法 | Lo: 本振 (\_\_init\_\_ 方法是一个特 Bw: 带宽 殊的方法,用于初始化类 nWin: 窗口数量? 的实例)中被用于设置实 LG 和 RG: 左右旋增益,两个 属性都被初始化为 1.0 centFreq: 中心频率,初始化 为 0.0 一系列空列表 Record: 记录时间, lonx, loffx,:左旋开关的功率? ronx, roffx:右旋开关的功率?

│这个类的目的是创建一个│ │输出: 类的实例属性

│数据对象,其中包含一系│ dataFile: 数据文件的路径

IData, rData:左右旋的数据 一些浮点数变量 |ImCal,rmCal:左右旋校准因子 lTsys, rTsys:左右旋系统温度 maxTm:最大积分时间? minTm:最小积分时间?

## def loadBkHead(self)

self.centFreq,使用

self.nWin 作为索引从

频率的值。

subFreq 中获取值,并加

上 self.Lo。可计算出中心

self.dataFile: 一个成员变量,表示 self.nWin: 类的一个成员变量,用作 索引来获取subFreq中的值。 标、观测带宽、通道带宽 self.Lo:类的一个成员变量,用作计 算self.centFreq的一部分。

dataFile: 数据文件的路径

rcvTy: 接收类型

Lo: 本振

Amp: 放大器的增益

(Low、High、No)

maxTm:最大积分时间?

minTm:最小积分时间?

创建一个存储子频率的空 输出: 列表。使用pyfits库打开 self.nchan: FITS文件头中提取的整数;通 | |指定路径的FITS文件从文 | | 道数。 件头中提取相关信息,包 self.crpix: FITS文件头中提取的浮点数;像 括频道数、参考像素坐 素中心的坐标。 self.obsbw: FITS文件头中提取的浮点数; 观测带宽(以GHz为单位) 将提取的信息赋值给类的 self.chbw: FITS文件头中提取的浮点数;通 成员变量。计算中心频率 道带宽(以GHz为单位)。 self.centFreq: 通过计算得到的中心频率。

#### def getRedData(self):

self.record: 类的成员变量;记录 self.ITsys: 类的成员变量;左旋系统温度 self.lData:类的成员变量;左旋数据 self.lmCal: 类的成员变量;左旋校准数据 self.rTsys:类的成员变量;右旋系统温度 self.rData:类的成员变量;右旋数据 self.rmCal: 类的成员变量;右旋校准数据

将类的成员变量组织成 一个包含三个元组的元 组。第一个元组包含了 包含了左旋相关数据的 Tsys、Data 和 mCal。 相关数据的 Tsys、Data 和 mCal。返回这个包含 三个元组的元组作为方

返回一个元组,其中包含三个元组。 | 第一个元组包含一个值,即self.record的值。 | 类的记录。第二个元组 | | 第二个元组包含三个值,分别是 | | self.lTsys、self.lData 和 self.lmCal 的值。 第三个元组包含三个值,分别是 第三个元组包含了右旋 self.rTsys、self.rData 和 self.rmCal 的值。

### def getTsys(self):

法的输出。

self.loffx, self.lonx: 左旋开关功 率,这些变量是类的成员变量; self.roffx, self.ronx: 右旋开关功 率,这些变量是类的成员变量; self.LG, self.RG: 左右旋增益,这 些变量也是类的成员变量

初始化四个空列表 (LOFFOFF, LOFFON, \ \ 输出: ROFFOFF, ROFFON左右旋开关频率?) | self.lTsys, self.rTsys: 这 用于存储数据。定义用于切片的索引 两个变量是类的成员 范围 (lp0, lp1, rp0, rp1),将原始数据 变量,表示左右旋的 切片成四个不同的数组。将切片得到 系统温度。 ││的数据组合到相应的列表(LOFFOFF 和 ││ LOFFON, ROFFOFF 和 ROFFON)。然后 计算它们的均值。继续计算 LOFFP 和 ROFFP,它们是 LOFFON 和 ROFFON 以及 LOFFOFF 和 ROFFOFF 的平均值。 最后用公式: self.lTsys = LOFFP \* self.LG; self.rTsys = ROFFP \* self.RG 左右旋的系统温度,分别赋值给 self.lTsys 和 self.rTsys。输出左右旋系

#### 统温度值。 def getCalData(self): 这段代码的目的是处理 输出: self.lData,

self.lonx, self.loffx: 左旋开 左右旋的数据。它首先 self.ronx, self.roffx: 右旋开 self.LG, self.RG: 左右旋的 self.lTsys, self.rTsys: 左右旋 的系统噪声温度。(有一 个条件语句判断如果 self.Amp 的值等于 No就会 将系统噪声温度重新设置 为1.0。)

self.rData: 经过处理和 计算左右旋开关数据的 校准的左右旋数据 平均值并乘以相应的增 | 益,然后通过左右旋数 据减去左右旋系统温度 对数据进行校准,如果 self.Amp 的值等于 No就 会将系统噪声温度重新 设置为 1.0。

## def getAvgBkData(self):

调用 self.getRawBkData() 输入:调用了 self.getRawBkData() 获取原始背景数据。然后 方法,获取原始背景(?)数据。 record: 表示记录或观测的序号或 标识。每个记录对应于一个数据 integ: 表示积分数据,即整个观测 过程中的积分值。 lon: 表示左旋开的数据。 loff: 表示左旋关的数据 ron: 表示右旋开的数据 roff: 表示右旋关的数据。 使用类的成员变量 self.Bw(频 宽)、self.chbw(通道宽度) self.crpix(通道中心索引) plt.plot 绘制左侧和右侧的

对原始背景数据的处 用带宽除以通道带宽得到 self.crpix计算积分通道中 ,通过 plt.plot 绘制了 中心位置)?。然后通过 通道数和中心通道计算出 积分通道范围的起始和结 束值。循环处理每个记 录,计算左右旋开关频率 数据的均值,需要前面得 到的索引范围并将其添加 到相应的列表中,将列表 转换为 NumPy 数组。使用

## 均值曲线,以进行可视 def getAvgTcalData(self):

self.crpix: 中心像素? self.centFreq: 中心频率

| 输入:调用了 self.getRawTcalData() | 方法,获取原始 Tcal 数据。 使用类的成员变量, self.Bw(频 宽)、self.chbw(通道宽度)和 self.crpix(通道中心索引)。

调用 self.getRawTcalData() 获 | 输出:存储在类的成员 取原始 Tcal 数据。计算积分 | 变量 self.lmCal 和 通道数和积分中心通道像 素,然后通过通道数和中心 通道计算出积分通道范围的 起始和结束值。然后使用 NumPy 的 mean() 方法计算左 右旋的 Tcal 数据在指定积分 通道范围内的均值。将计算 得到的均值分别存储在类的

成员变量 self.lmCal 和

self.rmCal 中。

self.rmCal 中的左右旋的 Tcal 数据的均值。

# def getAvgGain(self):

self.Amp: 表示是否有放大 如果 self.Amp 为 'No',将左 │器。如果为 'No',将左右旋│ │右旋的增益设置为1.0。 self.lonx, self.loffx, self.ronx, self.roffx: 左右旋开关的频 率,用于计算增益。 self.lmCal, self.rmCal: 左右旋 的校准数据,用于将观测数 据转换为物理单位(定标)

首先,判断放大器的影响, 输出: self.LG, self.RG (左右旋的增益值, 计 算左右旋开频率减去左 如果self.Amp不为 'No',计算 右旋关的频率的差再除 以校准数据)的均值的 左右旋的增益。增益的计算 | 倒数并存储在类的成员 方式左右旋开频率减去左右 旋关的频率的差再除以校准 变量self.LG 和 self.RG 中 数据除以校准数据。然后计 算左侧和右侧增益的均值。 取左侧和右侧均值的倒数, 即计算逆增益。逆增益用于 | 后续的数据校正,以将观测

输出:

左右通道的背景数据的元组

pData 包含了 FITS 文件中第 6 (frecord, integ, lon, loff, ron, roff):

# def getRawBkData(self):

初始化变量 pData (列表:

数据转换为物理单位

self.dataFile: 存储数据的 self.nWin: 用于索引的窗 | 表: 用于存储积分值) self.maxTm, self.minTm: 时间范围的上下限

个扩展的数据)和 Integ(列 一个包含有效记录、积分值以及 ,print提示信息表明正在打 一开数据文件。然后打开 FITS 文件并获取以下数据 record = pData.field(0): 从 pData 中提取第 0 列(字段) 的数据,并将其存储在名为 record 的变量中。这个变量 用于存储记录的时间信息? frecord = []: 初始化一个空列 表 frecord,这个列表将用于 存储有效记录的时间信息? row = len(pData):获取

pData 的行数,即数据表中记 录的数量,将其存储在变量 lon = []; loff = []; ron = []; roff = []: 初始化四个空列 表,分别用于存储左右旋开 关的频率。 然后遍历数据行,提取左右 旋的背景数据(IO、L1...)以 及左右旋的**频单**信息(IOF、 l1F)。计算时间差和积分值 通过左右旋频率计算得到。 当前循环迭代中处理的记录

record[idx]=record[idx] + deltTm(时间差)然后过滤 掉超出时间范围的数据。 录有效的记录和左右旋的数 据: Id0 = I0 \* 1.0 / IOF \* 1e-4。处理可能的异常值。存储 左右旋的开关的数据: lon.append(ld0)。返回结果

self.Amp: 类的成员变量, 一一行的数据提取出来。数据处 │表示放大器的影响,可能 ││理:将每行数据拆分为一组 为 'High'、'Low' 或 'No'。 self.nchan: 类的成员变量, self.centFreq: 类的成员变 self.crpix: 类的成员变量, self.chbw: 类的成员变量,

输入: 表示通道的数量。 量,表示中心频率。 表示中心像素。 表示通道带宽。

(frecord, integ, lon, loff, ron, def getRawTcalData(self): 打开文件:通过 输出:ltCal: 左旋在不同频 │self.rcvTy: 类的成员变量, ││open(self.rcvTy + '.dat', 'r') 打 ││率下的温度标定值。 │作为文件名的一部分。通 ││开以 self.rcvTy 命名的文件, │ rtCal: 右旋在不同频率下的 │过 self.rcvTy + '.dat' 构造文 ││并将文件对象存储在 File 变 ││温度标定值。 件名,并尝试打开该文 量中。读取文件内容:循环 | 迭代文件的每一行,将每一 ││数据,然后将这些数据分别 添加到不同的列表中 (f: 存储频率数据 ││Lh:存储高放大器模式下左 旋的校准值 ║: 存储低放大器模式下左旋 的校准值 │ rh:存储高放大器模式下右 rl: 存储低放大器模式下右旋 In: 存储无放大器模式下左 rn:存储无放大器模式下旋 通道的校准值)。 创建NumPy数组: 使用 NumPy 将列表转换为数组, 分别为x、y、z。根据条件插 值:根据 self.Amp 的值,选 |择使用 lh、ll 或 ln 进行插

> 值,得到两个插值函数 flm 和 frm。然后创建频率数组: │使用 w 计算频率数组。对频 率数组进行插值: 使用插值 函数 flm 和 frm 对频率数组进 行插值,得到 ltCal 和 rtCal。

def residuals(p,x,z):

类的实例变量,以便在

后续的代码中使用。

class MergeData:

mergeData.py

│这个函数的主要目的是计 ││输出: | 算拟合后的值与实际观测 | | 残差, 即观测数据 z 与通 │数据 z 之间的差异。在拟 ││过拟合模型 fun 计算得到 | 合问题中,优化算法通常 | | 的预测值之间的差异。 试图通过调整参数 p 以最 小化这些残差。因此, residuals 函数通常被用作 最小二乘法等优化算法的 目标函数。在最小二乘法 中,优化的目标是最小化 残差的平方和。

residuals=z-fun(x,p)

Fg: 一个表示条件的参 数,用于筛选数据,fg = datalist[6] 这一行代码 是从 datalist 列表中提 取第7个元素,并将其 赋值给变量 fg。

def \_\_init\_\_(self,antF,dtF,rcvTy,Amp,Lo,Bw,Sta,\ Freq,telF = "BB",multibeam = False):

|定义了一个MergeData类| 输出: 通过构造函数 \_\_init\_\_\_, antF: 天线文件? 将输入的参数值分配给 dtF: 数据文件? rcvTy:接收机类型。 Amp: 放大器增益。 Lo: 本振。 Bw: 带宽。 Sta: 站点。 Freq: 频率。 telF:望远镜文件? multibeam: 是否是多 波束观测。

getMergeData 方法的主要 输出:通过调用 功能是调用类中的 fitData self.fitData() 方法的返回值

## def fitData(self):

def getMergeData(self):

方法,并返回其结果。

调用了 self.mergeTelBk() 方 输出: |使用了 self.mergeTelBk() 方 │ │法,获取了一些数据,包 │ │返回一个包含三个元组的元 括方位角 (mAz)、俯仰角 组: (rawData, lFit, rFit)。 | (mEI)、记录 (record)、射电 | | rawData: 包含了一系列原始 | 的天文和射电望远镜的数 <sup>│</sup>│垂直角度 (aV),左右旋的 ││据,包括方位角 (mAz)、俯仰│ | | (ITsys、IData、ImCal 和 垂直角度 (aV),左右旋的 (ITsys、IData、ImCal 和 rTsys、rData、rmCal)。 法对左右旋的数据进行了 拟合,得到拟合结果。

││角 (mEl)、记录 (record)、射 rTsys、rData、rmCal)。根 | 电望远镜的水平角度 (aH)、 据站点信息(self.Sta)选 垂直角度 (aV),左右旋的 择了合适的数据作为拟合 (ITsys、IData、ImCal 和 │的输入,例如如果站点是 │ rTsys、rData、rmCal)。 □"AZ",则选择了 aH 作为 □ □ IFit:包含了与左旋相关的拟 fx。使用 self.subFitData 方 | 合信息,包括 IRMS(误差) 、Ihpbw(半功率波束宽度) 、lpter(主瓣峰值电平?) 构建了三个元组,包含了 、Ita(副瓣峰值电平?) 原始数据、左右旋的拟合 、IfitData(拟合后的左旋数 数据最终返回这个包含三 个元组的元组 ┘ │rFit: 包含了与右旋相关的拟 │ 合信息,包括 rRMS、rhpbw、rpter、rta、 fitData

def subFitData(self,px,py,pol): 计算标准差 sigma,通过一系列 │ 输出: px:水平轴的输入数据。 | 常数和类实例中的频率属性计 | 返回一个包含五个值的元组 (rms, py: 垂直轴的输入数据。 | 算而得。将输入数据 px 和 py | hpbw, pc, ta, fitd)。 转换为 NumPy 数组。print输出 rms:均方根误差。 |输入的 px 和 py。初始化全局 | hpbw: 半功率波束宽度。 变量 OFF 为 0。根据条件修改 pc: 波束中心的位置。? |全局变量 OFF 的值,具体而 | | ta: 波束主瓣的峰值电平。? | 言,如果 rcvTy 为 "Q",且 | | fitd:通过拟合得到的数据。 multibeam 为 True, pol 为 'L', 且 Sta 为 "AZ",则将 OFF 设置 为 84.0。类似地,如果 rcvTy 为 "K",且 multibeam 为 True, pol 为 'L',且 Sta 为 "AZ",则将 OFF 设置为 150.0 根据条件修改 px 的值,如果 Sta 为 "AZ",且 multibeam 为 True,且 pol 为 'L',则将 px 的 值加上 OFF。定义初始拟合参 数 Init(其中包括零阶项、一阶 项、py 的最大值、零点处的偏 移,以及之前计算得到的 Sigma)。调用 leastsq 函数进 行非线性最小二乘拟合,得到 拟合结果 r。然后计算均方根误 差 rms:对平均平方残差取平 方根计算得到均方根误差。波 東主瓣峰值电平 ta: 从拟合结 果r中提取第三个元素、波束 中心位置 Pc:从拟合结果 r 中提 取第四个元素、半功率波束宽 度 Hpbw:从拟合结果 r 中提取 第五个元素,即拟合的高斯函 数的标准差。乘以 2.35, 以将 标准差转换为高斯函数的半功 率波束宽度。math.fabs() 用于

获取绝对值,确保宽度为正值 。调用函数 fun(px, r) 获取通过 拟合得到的数据。print与拟合 结果相关的信息。返回包含拟 合结果的元组 (rms, hpbw, pc,

型,默认为"BB"。

def mergeTelBk(self): 录。创建 TelRawData 类的实 平和垂直数据进行时间插值。 数据、获取平均增益、获取 Tsys、获取校准数据以及获取修 │正数据。将校准结果的时间、左 右旋的 Tsys、左右旋的数据、左 右旋的天线校准、水平数据、垂 直数据添加到相应的列表中。将 各个列表的第一个元素提取出

来,形成一个完整的数据元组。 返回包含合并后数据的元组。

│初始化用于存储数据的列表,包││输出: 括左右旋数据、左右旋校准数 | 返回一个包含合并后数 │据、天线角度、左右旋Tsys、记 ││据的元组,包括: | meanAz: 方位角均值。 例,获取天文观测数据,包括时 meanEl: 俯仰角均值。 间 (Tm)、水平数据 (dH)、垂直数 | ftm: 合并后的时间数据 据 (dV)、方位角 (Az) 和俯仰角 faH: 水平角度。 (EI)。计算观测数据的最大和最 faV: 垂直角度。 小时间并使用 interp1d 函数对水 (flTsys, fldata, flmCal):处 理整合后的左旋系统温 创建 DIBASRawData 类的实例, 度、数据、校准数据。 获取射电望远镜校准数据。执行 (frTsys, frdata, frmCal):处 一系列方法,包括加载头文件、 理整合后的右旋系统温 获取平均Bk数据、获取平均 Tcal 度、数据、校准数据。