ICS 17.200

N 10

T/CSMT

团　　体　　标　　准

T/CSMT-DE-00*—20XX

免疫分析 测量结果的不确定度评定

Immunoassay　Uncertainty evaluation of measurement results

(征求意见稿)

XXXX - XX - XX 发布　　XXXX - XX - XX 实施

中国计量测试学会　　发布

目录

前　　言

本标准按照 GB/T 1.1-2020 给出的规则起草。

本标准由中国计量测试学会提出并归口。

本标准起草单位：中国计量科学研究院，江苏省农业科学院，新产业生物医学工程股份有限公司，贝克曼库尔特商贸(中国)有限公司，北京义翘神州科技股份有限公司，山东立菲生物产业有限公司

本标准主要起草人：武利庆，伦姚，高美静，王文峰，赵恒伟，杨滨，赵超，罗世闻，王晶，李浩正，温明达，张宁

免疫分析 测量结果的不确定度评定

1 范围

本文件规定了采用马尔可夫链蒙特卡罗法确定线性拟合、四参数模型拟合以及三次样条插值拟合确定免疫分析结果及其不确定度的评定方法。

本文件适用于标准溶液和仪器响应均存在不可忽略的不确定度时线性拟合、四参数模型拟合以及三次样条插值拟合确定免疫分析结果及其不确定度。

本文件不适用于医学检验实验室免疫分析结果的不确定度评定。

2 引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 27418-2017 测量不确定度评定和表示

GB/T 27419-2018 测量不确定度评定和表示 补充文件1: 基于蒙特卡洛方法的分布传播

GB/Z 43280-2023 医学实验室 测量不确定度评定指南

3 术语和计量单位

下列术语和定义适用于本文件

3.1

不确定度 uncertainty

测量不确定度 measurement uncertainty; uncertainty of measurement

利用可获得的信息，表征赋予被测量量值分散性的非负参数。

注1:测量不确定度包括由系统效应引起的分量，如与修正量和测量标准所赋量值有关的分量及定义的不确定度。有时对估计的系统效应未作修正，而是当作不确定度分量处理。

注2:此参数可以是诸如称为标准测量不确定度的标准差(或其特定倍数),或是说明了包含概率的区间半宽度。

注3:测量不确定度一般由若干分量组成。其中一些分量可根据一系列测量值的统计分布，按测量不确定度的A类评定进行评定，并可用标准差表征。而另一些分量则可根据经验或其他信息所获得的概率密度函数，按测量不确定度的B类评定进行评定，也用标准差表征。

注4:通常，对于一组给定的信息，测量不确定度是相应于所赋予被测量的值的，该值的改变将导致相应的不确定度的改变。

3.2

标准不确定度 standard uncertainty

以标准差表示的测量不确定度。

3.3

合成标准不确定度 combined standard uncertainty

由在一个测量模型中各输入量的标准测量不确定度获得的输出量的标准测量不确定度。

3.4

扩展不确定度 expanded uncertainty

合成标准测量不确定度与一个大于1的数字因子的乘积。

3.5

蒙特卡洛方法 Monte Carlo method

通过从概率分布中随机抽样而进行分布传播的方法。

3.6

贝叶斯定理 Bayes' Theorem

基于先验知识和新证据更新假设概率的核心公式，表示为：后验概率 ∝ 先验概率 × 似然函数。它通过数据调整初始信念（先验），结合观测数据的可能性（似然），计算出更新后的概率（后验），广泛应用于统计推断、机器学习和决策分析中。

3.7

马尔可夫链蒙特卡罗法 Markov Chain Monte Carlo

基于马尔可夫链的随机采样技术，通过构造平稳分布为目标概率分布的马尔可夫链，生成近似样本，用于估计复杂分布的积分、期望或进行贝叶斯推断。其核心是通过迭代生成样本，使链逐渐收敛到目标分布，结合蒙特卡罗积分实现高效近似计算。

4 概述

免疫分析是一种基于抗体和抗原之间特异性相互作用的生化分析技术，用于定性和定量检测样本中的特定靶标（如小分子、蛋白质、激素、细胞、病原体等）。该技术利用抗体高度的特异性和结合能力识别并结合到靶标上，通过各种信号转换方法将这种结合事件转化为可测量的信号，从而实现对靶标的定性和定量检测。免疫分析时生物标准溶液和仪器响应通常都具有不可忽略的不确定度，同时数据处理也具有复杂性，其结果计算的数学模型存在多种类型，经常使用的数据处理方式包括线性拟合、双对数线性拟合、四参数模型拟合以及三次样条插值拟合等。虽然不同的数据处理方式对应不同的测量模型，但是输入量及输入量的不确定度是相同的，输入量包括系列标准溶液的浓度x_i和在测量仪器上的响应值y_i，以及试样在测量仪器上的响应值y；输入量的不确定度包括系列标准溶液浓度的不确定度u（x_i）和在测量仪器上响应值的不确定度u（y_i），以及试样在测量仪器上响应值的不确定度u（y）。拟合过程就是通过输入量（x_i，y_i）建立起拟合函数$y=F$（x）的过程，根据试样在测量仪器上的响应值y计算试样浓度就是根据函数F的逆函数$x=F^{-1}$（y）计算x。实际上F^{-1}（y）中的各个系数是通过输入量（x_i，y_i）通过一定算法得到的，如果以原始的（x_i，y_i）和y_0作为输入量，F^{-1}（y_0）可以改写为$x=S$（x_i，y_i，y_0），因此根据不确定度传播率x_0的方差可以通过式（1）计算：

$$u_{x_0}^2 = \left(\frac{\partial x_0}{\partial y_0}\right)^2 u_{y_0}^2 + \nabla_\theta x_0^T \cdot \Sigma_\theta \cdot \nabla_\theta x_0 + (\text{可能来自输入数据} u_{x_i}, u_{y_i}) \tag{1}$$

式中，

y_0 —— 试样的仪器响应值；

x_0 —— 试样分析结果；

u_{x_0} —— x_0的不确定度；

$\nabla_\theta x_0$ —— x_0对模型参数θ的梯度向量；

Σ_θ —— 模型参数θ的协方差矩阵；

u_{x_i} —— 标准溶液浓度x_i的不确定度；

u_{y_i} —— 浓度x_i的标准溶液仪器响应y_i的不确定度。

分别采用线性拟合、双对数线性拟合、四参数模型拟合以及三次样条插值拟合时函数S（x_i，y_i，y_0）的具体形式不一样，但计算方式都是一样的。另外，由于函数S（x_i，y_i，y_0）的具体形式可能很复杂，通常编程采用数值法求解。

5 免疫分析结果不确定度的评定

5.1 免疫分析的一般过程

在进行免疫分析时，一般首先通过标准物质稀释配制一系列标准溶液共 q 个（即 i=1, 2，... ，q），其浓度分别为 x_i，相应的不确定度为 u（x_i）；通过仪器对标准溶液和试样进行重复分析，得到各个浓度标准溶液的仪器响应值 y_i 以及试样的仪器响应值 y，相应的不确定度分别为 u（y_i）和 u（y）。通过（x_i，y_i）建立起拟合函数 y=F（x），根据函数 F 的逆函数 x=F^{-1}（y）将试样在测量仪器上的响应值 y 代入到 F^{-1}（y）中计算出试样浓度 x。

5.2 $u(x_i)$的计算

u（x_i）是各个水平标准溶液的不确定度，一般通过标准物质逐级稀释得到，稀释到 x_i 浓度时的不确定度由标准物质的不确定度和稀释过程中使用的移液器的不确定度传播得到。其中由移液器引起的相对不确定度 $u_{d,r}$ 根据式（2）计算。

$$u_{d,r}=\sqrt{\sum_{j=1}^{l}\left(\frac{u_{V,j}}{V_j}\right)^2} \tag{2}$$

式中，

$u_{V,j}$——第 j 次移液时使用的移液器的标准不确定度；

V_j——第 j 次移液时使用移液器移取的溶液体积；

l—— 稀释到 x_i 浓度时移取液体的总次数。

当移液器附带校准证书时 $u_{d,r}$ 根据式（3）计算。

$$u_{V,j} = \frac{U_{V,j}}{k} \tag{3}$$

式中，

$U_{V,j}$——第 j 次移液时使用的移液器校准证书中提供的扩展不确定度；

k—— 移液器校准证书中提供的包含因子，通常 k=2。

当移液器附带检定证书时 $u_{d,r}$ 根据式（4）计算。

$$u_{V,j} = \frac{|MPE|}{\sqrt{3}} \tag{4}$$

式中，

MPE——检定规程中对应级别和量程给出的最大允许误差。

由标准物质引入的不确定度 u_{CRM} 根据式（5）计算。

$$u_{CRM} = \frac{U_{CRM}}{k_{CRM}} \tag{5}$$

式中，

U_{CRM} ——标准物质证书中提供的扩展不确定度；

k_{CRM} ——标准物质证书中提供的包含因子，通常 $k_{CRM} = 2$ 。

各个标准溶液浓度 x_i 的不确定度 u（x_i）根据式（6）计算：

$$u(x_i) = x_i\sqrt{\left(\frac{u_{CRM}}{x_0}\right)^2 + u_{d,r}^2} \tag{6}$$

式中，

x_0 ——证书中标准物质的浓度；

在没有标准物质的情况下如果可以采用计量方法确定母液的标准值和不确定度作为 x_0 和 u（x_0），再按照上述方式评定不确定度。在无法确定或忽略母液引入的不确定度时，逐级稀释到标准溶液 x_i 时的不确定度为：

$$u(x_i) = x_t u_{d,r} \tag{7}$$

5.3 $u(y_i)$和 $u(y)$的计算

每个标准溶液重复测定 p 次，如果 $p \geqq 6$ 则可以根据公式（8）计算 $u(y_i)$：

$$u(y_i) = \frac{1}{\sqrt{p}}\sqrt{\frac{\sum_{k=1}^{p}(A_i - \overline{A})^2}{p-1}} \tag{8}$$

式中，

A_i——仪器信号响应值；

$\overline{A}$ ——p 次测量仪器信号响应值的平均值。

当 $p<6$ 时则可以根据公式（9）计算 $u(y_i)$：

$$u(y_i) = \frac{|A_{\max} - A_{\min}|}{\sqrt{l}C} \tag{9}$$

式中，

A_{max}——测量仪器响应最大值；

A_{min}——测量仪器响应最小值；

C——相应测量次数下的极差系数。

试样分析时测定 m 个平行子样、每个子样重复测定 n 次，根据式（10）计算试样分析结果的的$u(y)$：

$$u(y)=\frac{1}{\sqrt{mn}}\sqrt{\frac{\sum_{s=1}^{m}\sum_{t=1}^{n}(A_{st}-\bar{\bar{A}})^2}{mn-1}} \tag{10}$$

式中，

A_{st}——第 s 个平行子样第 t 次测定时测量仪器响应值；

$\bar{\bar{A}}$——m 个平行子样每个子样重复测定 n 次时测量仪器响应值的算数平均值；

5.4 $u(x)$的计算

根据选定的线性拟合、双对数线性拟合、四参数拟合或三次样条插值拟合算法通过输入量（x_i，y_i）建立起拟合函数 $y=F$（x），根据试样在测量仪器上的响应值 y 和函数 F 的逆函数 $x=F^{-1}$（y）计算 x。F^{-1}（y）中的各个系数是通过输入量（x_i，y_i）通过一定算法得到的，如果以原始的（x_i，y_i）和 y 作为输入量，F^{-1}（y）可以改写为 $x=S$（x_i，y_i，y），根据不确定度传播率，x_0 的不确定度 u（x_0）可以通过式（11）计算

$$u_{x_0}=\sqrt{\left(\frac{\partial x_0}{\partial y_0}\right)^2 u_{y_0}^2+\nabla_{\theta}x_0^T\cdot\Sigma_{\theta}\cdot\nabla_{\theta}x_0+(\text{可能来自输入数据}u_{x_i},u_{y_i})} \tag{11}$$

由于函数 S（x_i，y_i，y）的具体形式复杂，因此采用编程的方式通过马尔可夫链蒙特卡罗法计算，示例见附录 A~D。

5.5 U 的计算

取扩展因子 k=2，则免疫分析结果的扩展不确定度通过式（12）计算。

$$U=ku_{x_0} \tag{12}$$

附录 A　线性回归免疫分析结果不确定度的评定示例

（资料性）

A.1、实验方法

采用商品化 Cry3A 转基因产品酶联免疫检测试剂盒高低两个水平的转基因试样进行检测，使用试剂盒自带的 6 个水平的标准溶液绘制标准曲线，每个浓度的标准溶液重复检测 2 次，每个试样重复检测 6 次。

A.2、实验结果

酶联免疫分析实验结果如表 A.1 所示。

表 A.1 Cry3A 试样酶联免疫分析试剂盒检测结果

样品名称	浓度 ng/mL	吸光度			吸光度平均值
空白	0	0.162	0.178	/	0.1700
标准 1	0.25	0.235	0.235	/	0.2350
标准 2	0.50	0.296	0.285	/	0.2905
标准 3	1.00	0.405	0.441	/	0.4230
标准 4	2.00	0.624	0.623	/	0.6235
标准 5	4.00	0.891	0.974	/	0.9325
试样 1	/	0.267	0.277	0.276	0.2650
		0.259	0.259	0.253	
试样 2	/	0.449	0.434	0.463	0.4480
		0.445	0.450	0.446	

A.3、$u(x_i)$, $u(y_i)$和 $u(y)$的计算

u（x_i）是各个水平标准溶液的不确定度，一般通过标准物质逐级稀释得到，稀释到 x_i 浓度时的不确定度由标准物质的不确定度和稀释过程中使用的移液器的不确定度传播得到。其中由移液器引起的相对不确定度 $u_{d,r}$ 根据式（A.1）计算。

$$u_{d,r}=\sqrt{\sum_{j=1}^{l}\left(\frac{u_{V,j}}{V_j}\right)^2} \tag{A.1}$$

式中 $u_{V,j}$ 为第 j 次移液时使用的移液器的标准不确定度； V_j 为第 j 次移液时使用移液器移取的溶液体积；l 为稀释到 x_i 浓度时移取液体的总次数。

由标准物质引入的不确定度 u_{CRM} 根据式（A.2）计算。

$$u_{CRM}=\frac{U_{CRM}}{k_{CRM}} \tag{A.2}$$

式中，

U_{CRM} ——标准物质证书中提供的扩展不确定度；

k_{CRM} ——标准物质证书中提供的包含因子，通常 $k_{CRM}=2$。

其中移液器校准证书中取 500μL 时的扩展不确定度为 3.4μL。即 U_{CRM} =3.4 μL， u_{CRM} =1.7μL。

各个标准溶液浓度 x_i 的不确定度 u（x_i）根据式（A.3）计算：

$$u(x_i)=x_i\sqrt{\left(\frac{u_{CRM}}{x_0}\right)^2+u_{d,r}^2} \tag{A.3}$$

式中 x_0 为证书中标准物质的浓度；其中 x_0=500。

标准溶液每个测试 2 次，按照式（A.4）计算平均值的不确定度：

$$u(y_i)=\frac{|A_{\max}-A_{\min}|}{\sqrt{l}C} \tag{A.4}$$

式中 $A_{\max}$ 和 $A_{\min}$ 分别为两次吸光度测定结果，l 为测量次数，l=2；C 为极差系数，2 次测量的极差

系数为 1.128。

试样每个测试 6 次，按照式（A.5）计算平均值的不确定度：

$$u(y)=\frac{1}{\sqrt{mn}}\sqrt{\frac{\sum_{s=1}^{m}\sum_{t=1}^{n}(A_{st}-\bar{\bar{A}})^2}{mn-1}} \tag{A.5}$$

式中，m为子样数量，m=1；n为每个子样重复测量次数，n=6；A_{st}为子样s第t次吸光度测量结果，$\bar{\bar{A}}$为子样吸光度测量结果平均值。

根据上述计算公式，标准溶液和试样的$u(x_i)$,$u(y_i)$和$u(y)$计算过程如下。

标准溶液的 $u(x_i)$计算结果如下：

$$u(x_0)=x_0\sqrt{u_{d,r}^2}=4\sqrt{0.1^2}=0.4$$

$$u(x_4)=x_4\sqrt{8\times\left(\frac{1.7}{500}\right)^2+0.1^2}=2\sqrt{8\times\left(\frac{1.7}{500}\right)^2+0.1^2}=0.20023$$

$$u(x_3)=x_3\sqrt{6\times\left(\frac{1.7}{500}\right)^2+0.1^2}=1\sqrt{6\times\left(\frac{1.7}{500}\right)^2+0.1^2}=0.10023$$

$$u(x_2)=x_2\sqrt{4\times\left(\frac{1.7}{500}\right)^2+0.1^2}=0.5\sqrt{4\times\left(\frac{1.7}{500}\right)^2+0.1^2}=0.05017$$

$$u(x_1)=x_1\sqrt{2\times\left(\frac{1.7}{500}\right)^2+0.1^2}=0.25\sqrt{2\times\left(\frac{1.7}{500}\right)^2+0.1^2}=0.02512$$

标准溶液的 $u(y_i)$计算结果如下：

$$u(y_0)=\frac{|0.178-0.162|}{1.128\times\sqrt{2}}=0.01003$$

$$u(y_1)=\frac{|0.235-0.235|}{1.128\times\sqrt{2}}=0.00000$$

$$u(y_2)=\frac{|0.296-0.285|}{1.128\times\sqrt{2}}=0.006896$$

$$u(y_3)=\frac{|0.441-0.405|}{1.128\times\sqrt{2}}=0.02257$$

$$u(y_4) = \frac{|0.624 - 0.623|}{1.128 \times \sqrt{2}} = 0.0006269$$

$$u(y_5) = \frac{|0.974 - 0.891|}{1.128 \times \sqrt{2}} = 0.05203$$

试样的$u(\overline{y})$计算结果如下：

$$u(\overline{y_1}) = \frac{1}{\sqrt{6}}\sqrt{\frac{0.002^2 + 0.012^2 + 0.011^2 + (-0.006)^2 + (-0.006)^2 + (-0.012)^2}{5}} = 0.004020$$

$$u(\overline{y_2}) = \frac{1}{\sqrt{6}}\sqrt{\frac{0.001^2 + (-0.014)^2 + 0.015^2 + (-0.003)^2 + 0.002^2 + (-0.002)^2}{5}} = 0.003825$$

根据上述计算方式，标准溶液和试样的$u(x_i)$,$u(y_i)$和$u(y)$计算结果见表A.2。

表A.2　标准溶液和试样的$u(x_i)$,$u(y_i)$和u（y）计算结果

样品名称	x_i	$u(x_i)$	y_i	$u(y_i)$	$\overline{y}$	$u(\overline{y})$
空白	0	0	0.1700	0.01003	/	/
标准 1	0.25	0.02512	0.2350	0	/	/
标准 2	0.50	0.05017	0.2905	0.006896	/	/
标准 3	1.00	0.1002	0.4230	0.02257	/	/
标准 4	2.00	0.2002	0.6235	0.0006269	/	/
标准 5	4.00	0.4000	0.9325	0.05203	/	/
样品 1	/	/	/	/	0.2650	0.004020
样品 2	/	/	/	/	0.4480	0.003825

A.4、$u(x)$和 U 的计算

通过Python编程计算u（x），代码如下：

```
import numpy as np

import matplotlib
```

```
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
import emcee
from scipy.stats import norm
# 数据
x_data = np.array([0, 0.25, 0.50, 1.00, 2.00, 4.00])
y_data = np.array([0.17, 0.235, 0.2905, 0.423, 0.6235, 0.9325])
u_x_data = np.array([1e-6, 0.02512, 0.05017, 0.1002, 0.2002, 0.4000])
u_y_data = np.array([0.010029883, 1e-6, 0.006895545, 0.022567238, 0.000626868, 0.05203002])
y0 = 0.2650
u_y0 = 0.004020
# 定义线性模型
def linear_model(x, a, b):
    return a * x + b
# 定义对数似然函数 (考虑 x 和 y 不确定度)
def log_likelihood(theta, x, y, yerr, xerr):
    a, b, log_f = theta
    model = linear_model(x, a, b)
    sigma2 = yerr**2 + (a * xerr)**2 + model**2 * np.exp(2 * log_f)  # 包含 x 不确定度
    return -0.5 * np.sum((y - model)**2 / sigma2 + np.log(sigma2))
# 定义先验分布
def log_prior(theta):
    a, b, log_f = theta
    if -5.0 < a < 0.5 and -0.5 < b < 1.0 and -10 < log_f < 1.0:
        return 0.0
    return -np.inf
```

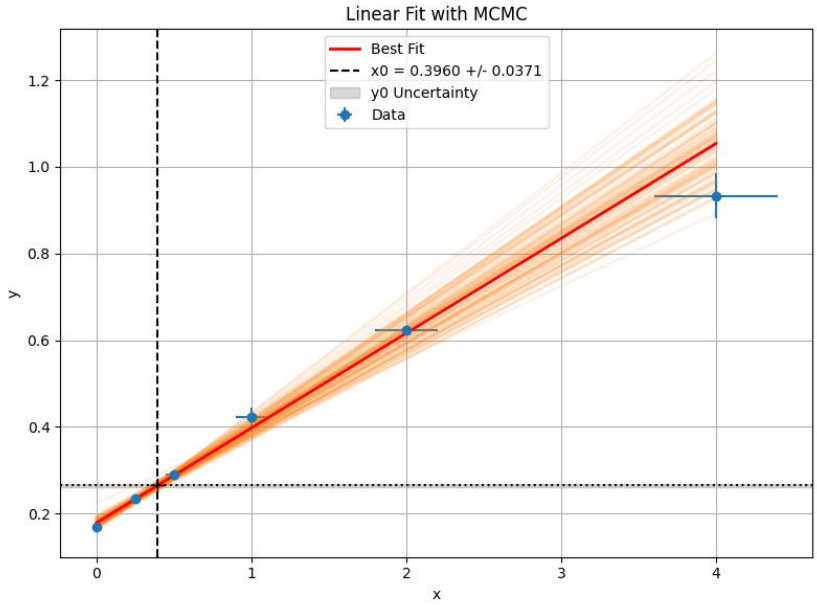
```
# 定义后验分布
def log_probability(theta, x, y, yerr, xerr):
    lp = log_prior(theta)
    if not np.isfinite(lp):
        return -np.inf
    return lp + log_likelihood(theta, x, y, yerr, xerr)
# MCMC 设置
nwalkers = 32
ndim = 3  # a, b, log_f
nsteps = 5000
# 初始位置
initial = np.array([0.1, 0.2, -3])  # a, b, log_f 的初始值
pos = initial + 1e-4 * np.random.randn(nwalkers, ndim)
# 创建采样器
sampler = emcee.EnsembleSampler(nwalkers, ndim, log_probability, args=(x_data, y_data, u_y_data, u_x_data))
# 运行 MCMC
sampler.run_mcmc(pos, nsteps, progress=True)
# 获取样本
samples = sampler.get_chain(flat=True)
# 提取 a 和 b 的样本
a_samples = samples[:, 0]
b_samples = samples[:, 1]
# 考虑 u_y0 的影响：对 y0 进行抽样
n_y0_samples = 1000
y0_samples = np.random.normal(y0, u_y0, n_y0_samples)
```

```
# 初始化 x0 样本
x0_samples = np.array([])


# 对每个 y0 样本，计算 x0 的分布
for y0_sample in y0_samples:
    x0_samples = np.concatenate([x0_samples, (y0_sample - b_samples) / a_samples])
# 计算 x0 的不确定度
x0_mean = np.mean(x0_samples)
x0_std = np.std(x0_samples)
print(f"x0 = {x0_mean:.4f} +/- {x0_std:.4f}")
# 可视化
# 1. MCMC 采样轨迹
fig, axes = plt.subplots(ndim, figsize=(10, 7), sharex=True)
labels = ["a", "b", "log_f"]
for i in range(ndim):
    ax = axes[i]
    ax.plot(sampler.get_chain()[:, :, i], "k", alpha=0.3)
    ax.set_xlim(0, len(sampler.get_chain()))
    ax.set_ylabel(labels[i])
axes[-1].set_xlabel("step number")
plt.suptitle("MCMC Sampling Trace")
plt.tight_layout(rect=[0, 0.05, 1, 0.95]) # 调整子图，避免标题重叠
plt.show()
# 2. 后验分布
fig, axes = plt.subplots(ndim, figsize=(10, 7))
labels = ["a", "b", "log_f"]
```

```
for i in range(ndim):
    ax = axes[i]
    ax.hist(samples[:, i], bins=50, density=True, color="steelblue", edgecolor="black")
    ax.set_xlabel(labels[i])
    ax.set_ylabel("Probability Density")
plt.suptitle("Posterior Distributions")
plt.tight_layout(rect=[0, 0.05, 1, 0.95])
plt.show()
# 3. x0 的后验分布
plt.hist(x0_samples, bins=50, density=True, color="steelblue", edgecolor="black")
plt.xlabel("x0")
plt.ylabel("Probability Density")
plt.title("Posterior Distribution of x0")
plt.axvline(x=x0_mean, color='red', linestyle='--', label=f'Mean = {x0_mean:.4f}')
plt.axvline(x=x0_mean + x0_std, color='green', linestyle='--', label=f'Mean + Std = {x0_mean + x0_std:.4f}')
plt.axvline(x=x0_mean - x0_std, color='green', linestyle='--', label=f'Mean - Std = {x0_mean - x0_std:.4f}')
plt.legend()
plt.show()
# 4. 拟合图
plt.figure(figsize=(8, 6))
plt.errorbar(x_data, y_data, xerr=u_x_data, yerr=u_y_data, fmt="o", label="Data")
# 绘制样本的拟合线
nsamples_plot = 100
inds = np.random.randint(len(samples), size=nsamples_plot)
for ind in inds:
    a, b, log_f = samples[ind]
```

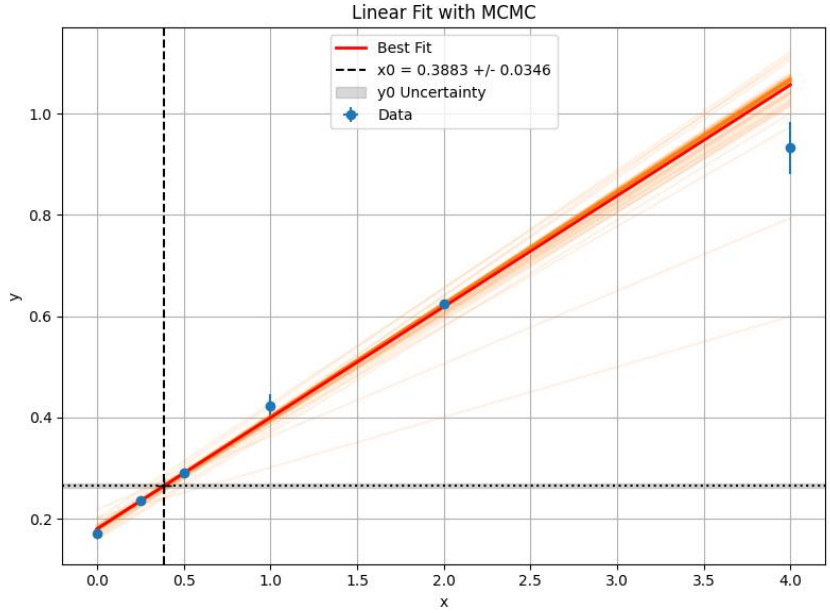
```
        y_fit = linear_model(x_data, a, b)
        plt.plot(x_data, y_fit, "C1", alpha=0.1)
    # 绘制最佳拟合线
    a_best, b_best, log_f_best = np.mean(a_samples), np.mean(b_samples), np.mean(samples[:,2])  # 使用平均值作为最佳估计
    x_fit = np.linspace(np.min(x_data), np.max(x_data), 100)
    y_fit = linear_model(x_fit, a_best, b_best)
    plt.plot(x_fit, y_fit, "r-", linewidth=2, label="Best Fit")
    # 标注 x0 位置
    plt.axvline(x=x0_mean, color="k", linestyle="--", label=f"x0 = {x0_mean:.4f} +/- {x0_std:.4f}")
    # 绘制 y0 的不确定区域
    plt.axhspan(y0 - u_y0, y0 + u_y0, color='gray', alpha=0.3, label='y0 Uncertainty')
    # 绘制 y0 的中心线
    plt.axhline(y=y0, color="k", linestyle=":")
    # 添加十字线
    plt.plot(x0_mean, y0, marker='+', color='k', markersize=10)
    plt.xlabel("x")
    plt.ylabel("y")
    plt.title("Linear Fit with MCMC")
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()
```

低值试样线性拟合结果如图 A.1 所示, 试样分析结果 $x = (0.3960 \pm 0.0371)$ ng/mL, 即 x 值的不确定 $u(x)$= 0.0371 ng/mL, 取扩展因子 k=2, 则扩展不确定度 U=2×0.0371=0.075 ng/mL, 试样分析结果可以表示为 (0.396 ± 0.075) ng/mL。



图A.1 低值试样线性拟合计算结果

如果使用试剂盒自带的标准溶液直接测定，这些标准溶液没有提供不确定度也无法通过分析确定不确定度，则忽略掉标准溶液的不确定度，将 $u(x_i)$ 均设置为 10^{-6} ng/mL，此时低值试样线性拟合结果如图 A.2 所示，试样分析结果 x=（0.3883 ± 0.0346）ng/mL，即 x 值的不确定 $u(x)$=0.0346 ng/mL，取扩展因子 k=2，则扩展不确定度 U=2×0.0346= 0.070 ng/mL，试样分析结果可以表示为（0.388 ± 0.070）ng/mL。



图A.2 低值试样线性拟合计算结果[$u(x_i)$ =0]

高值试样线性拟合结果如图 A.3 所示, 试样分析结果 $x=(1.2379\pm0.1314)$ng/mL, 即 x 值的不确定 $u(x)$= 0.131 ng/mL，取扩展因子 k=2，则扩展不确定度 U=2×0.131=0.27 ng/mL，试样分析结果可以表示为（1.23 ± 0.27）ng/mL。

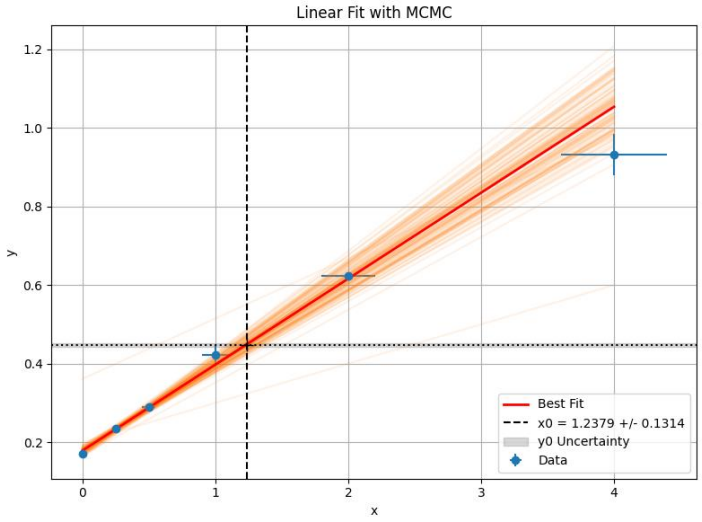
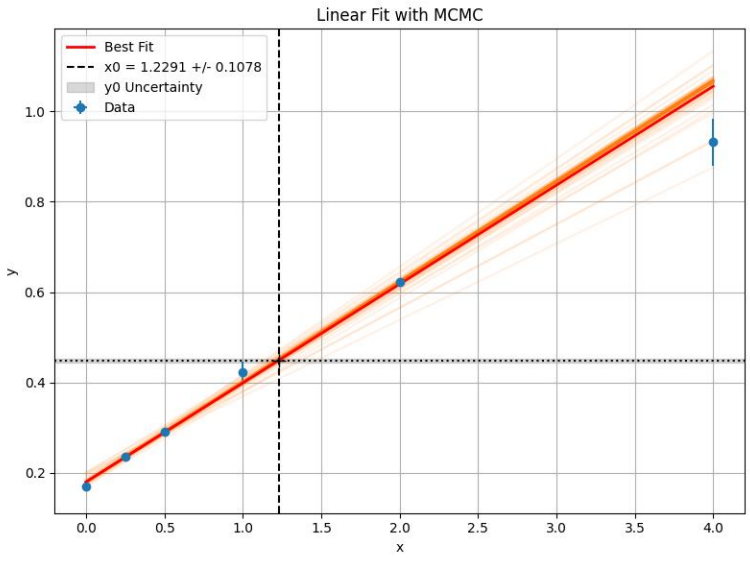


图 A.3 高值试样线性拟合计算结果

如果使用试剂盒自带的标准溶液直接测定，这些标准溶液没有提供不确定度也无法通过分析确定不确定度，则忽略掉标准溶液的不确定度，将 $u(x_i)$ 均设置为 10^{-6} ng/mL，此时高值试样线性拟合结果如图 A.4 所示，试样分析结果 x=（1.2291 ± 0.1078）ng/mL，即 x 值的不确定 $u(x)$=0.108 ng/mL，取扩展因子 k=2, 则扩展不确定度 U=2×0.108= 0.22 ng/mL, 试样分析结果可以表示为 (1.22 ± 0.22) ng/mL。



图A.4 高值试样线性拟合计算结果[$u(x_i)$ =0]

附录 B　四参数拟合免疫分析结果不确定度的评定示例

（资料性）

B.1、实验方法

同 A.1。

B.2、实验结果

同 A.2。

B.3、$u(x_i)$, $u(y_i)$和 $u(y)$的计算

同 A.3。

B.4、$u(x)$和 U 的计算

通过Python编程计算u（x），代码如下：

```
import numpy as np
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
import emcee
import corner
from scipy.optimize import curve_fit
from scipy.stats import norm


# Define the four-parameter model
def four_param_model(x, A, B, C, D):
    return D + (A - D) / (1 + (x/C)**B)
```

```
# Define the log-likelihood function
def log_likelihood(theta, x_data, y_data, u_x_data, u_y_data):
    A, B, C, D = theta
    model = four_param_model(x_data, A, B, C, D)
    sigma2 = u_y_data**2 + (B * (A - D) * (x_data**B) * u_x_data / (C**B * (1 + x_data/C)**B)**2))**2
    return -0.5 * np.sum((y_data - model)**2 / sigma2 + np.log(2 * np.pi * sigma2))

# Define the log-prior function
def log_prior(theta):
    A, B, C, D = theta
    if not all(0 < value < 10 for value in [A, B, C, D]):
        return -np.inf
    return 0.0

# Define the log-probability function
def log_probability(theta, x_data, y_data, u_x_data, u_y_data):
    lp = log_prior(theta)
    if not np.isfinite(lp):
        return -np.inf
    return lp + log_likelihood(theta, x_data, y_data, u_x_data, u_y_data)

# Define the data
x_data = np.array([0, 0.25, 0.50, 1.00, 2.00, 4.00])
y_data = np.array([0.17, 0.235, 0.2905, 0.423, 0.6235, 0.9325])
u_x_data = np.array([1e-6, 0.02512, 0.05017, 0.1002, 0.2002, 0.4000])
```

```
u_y_data = np.array([0.010029883, 1e-6, 0.006895545, 0.022567238, 0.000626868, 0.05203002])

y0 = 0.2650

u_y0 = 0.004020


# Initial guess for the parameters

initial_guess = [0.169915, 1.064054, 6.244783, 2.156547]    # A, B, C, D


# Maximum likelihood estimation (MLE) using curve_fit

try:

    popt, pcov = curve_fit(four_param_model, x_data, y_data, p0=initial_guess, sigma=u_y_data,
absolute_sigma=True)

    A_fit, B_fit, C_fit, D_fit = popt

    print("MLE Fit Parameters (curve_fit):")

    print("A =", A_fit)

    print("B =", B_fit)

    print("C =", C_fit)

    print("D =", D_fit)

except RuntimeError as e:

    print(f"Error during curve_fit: {e}")

    A_fit, B_fit, C_fit, D_fit = initial_guess    # Fallback to initial guess

    pcov = np.eye(4) * 1e-6    # Dummy covariance matrix


# MCMC setup

nwalkers = 32

ndim = 4

nsteps = 10000
```

```
burnin = 2000


# Initialize the walkers
rng = np.random.default_rng()
initial = np.array(initial_guess)
pos = initial + 1e-4 * rng.standard_normal((nwalkers, ndim))


# Create the sampler
sampler = emcee.EnsembleSampler(nwalkers, ndim, log_probability, args=(x_data, y_data, u_x_data, u_y_data))


# Run the MCMC sampler
sampler.run_mcmc(pos, nsteps, progress=True)


# Discard the burn-in samples
samples = sampler.get_chain(discard=burnin, flat=True)


# Analyze the MCMC results
A_mcmc, B_mcmc, C_mcmc, D_mcmc = np.mean(samples, axis=0)
print("\nMCMC Fit Parameters:")
print("A =", A_mcmc)
print("B =", B_mcmc)
print("C =", C_mcmc)
print("D =", D_mcmc)


# Corner plot
```

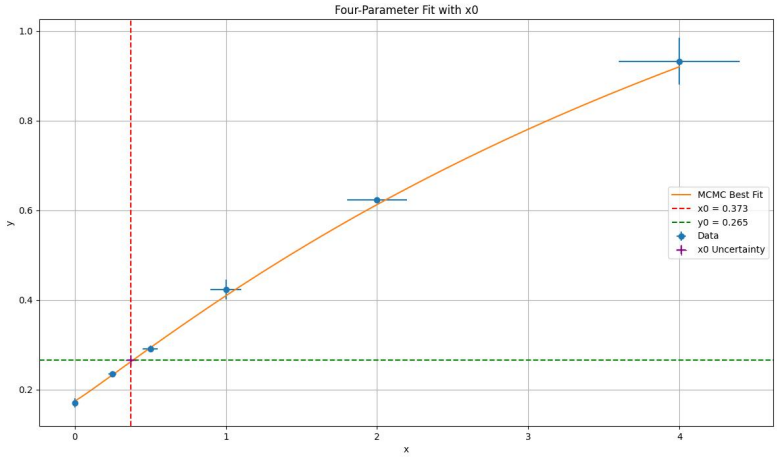
```
corner.corner(samples, labels=["A", "B", "C", "D"], truths=[A_fit, B_fit, C_fit, D_fit])
plt.suptitle("MCMC Parameter Distributions", y=0.98)
plt.show()    # Display the corner plot
plt.close()


# Function to find x0 and its uncertainty, now including u_y0
def find_x0(y0, u_y0, samples, x_data, num_y0_samples=100):
    x0_values = []
    A_samples = samples[:, 0]
    B_samples = samples[:, 1]
    C_samples = samples[:, 2]
    D_samples = samples[:, 3]
    rng = np.random.default_rng()    # Use a random number generator
    for A, B, C, D in zip(A_samples, B_samples, C_samples, D_samples):
        # Sample y0 values from a normal distribution
        y0_samples = norm.rvs(loc=y0, scale=u_y0, size=num_y0_samples, random_state=rng)
        for y0_sampled in y0_samples: #Iterate over the sampled y0 values
            # Use a root-finding method to solve for x0
            def equation(x):
                return four_param_model(x, A, B, C, D) - y0_sampled
            # Providing a bracket around the root can help the solver
            x_lower = min(x_data)
            x_upper = max(x_data)
            try:
                from scipy.optimize import brentq
                x0 = brentq(equation, x_lower, x_upper)
```

```
                x0_values.append(x0)
            except ValueError:
                # If brentq fails (e.g., no root within the bracket), return NaN
                x0_values.append(np.nan)
            except Exception as e:
                print(f"Error during root finding: {e}")
                x0_values.append(np.nan)
    x0_values = np.array(x0_values)
    x0_values = x0_values[~np.isnan(x0_values)]    # Filter out NaN values
    if len(x0_values) == 0:
        return np.nan, np.nan    # Return NaN if no valid x0 values are found
    x0_mean = np.mean(x0_values)
    x0_uncertainty = np.std(x0_values)
    return x0_mean, x0_uncertainty

# Find x0 and its uncertainty
x0, ux0 = find_x0(y0, u_y0, samples, x_data) #Pass u_y0 to find_x0
print("\nx0 =", x0)
print("ux0 =", ux0)

# Plot the data and the fit
plt.figure(figsize=(10, 8))
plt.errorbar(x_data, y_data, xerr=u_x_data, yerr=u_y_data, fmt="o", label="Data")

# Plot the best-fit curve
x_fit = np.linspace(min(x_data), max(x_data), 100)
```

```
y_fit = four_param_model(x_fit, A_mcmc, B_mcmc, C_mcmc, D_mcmc)
plt.plot(x_fit, y_fit, label="MCMC Best Fit")

# Plot x0 with crosshairs
if not np.isnan(x0):
    plt.axvline(x=x0, color='red', linestyle='--', label=f'x0 = {x0:.3f}')
    plt.axhline(y=y0, color='green', linestyle='--', label=f'y0 = {y0:.3f}')
    plt.errorbar(x0, y0, xerr=ux0, yerr=u_y0, fmt='+', color='purple', markersize=12, label='x0 Uncertainty')
plt.xlabel("x")
plt.ylabel("y")
plt.title("Four-Parameter Fit with x0")
plt.legend()
plt.grid(True)
plt.show()    # Display the fit plot
plt.close()

# Plot MCMC samples
plt.figure(figsize=(12, 8))
for i in range(ndim):
    plt.subplot(ndim, 1, i + 1)
    plt.plot(sampler.get_chain()[:, :, i], color="k", alpha=0.3)
    plt.ylabel(f"Parameter {['A', 'B', 'C', 'D'][i]}")
plt.xlabel("Step Number")
plt.suptitle("MCMC Sample Traces", y=0.98)
plt.tight_layout()
plt.show() # Display MCMC traces
```

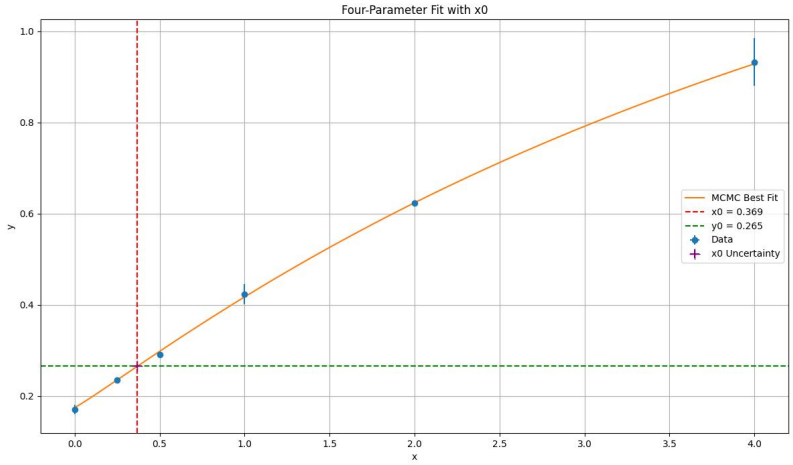
```
plt.close()

# Plot posterior distributions

plt.figure(figsize=(12, 8))

for i in range(ndim):

    plt.subplot(ndim, 1, i + 1)

    plt.hist(samples[:, i], bins=50, density=True, alpha=0.6, color="skyblue")

    plt.xlabel(f"Parameter {['A', 'B', 'C', 'D'][i]}")

    plt.ylabel("Density")

    plt.title(f"Posterior Distribution of Parameter {['A', 'B', 'C', 'D'][i]}")

plt.suptitle("Posterior Distributions of Parameters", y=0.98)

plt.tight_layout()

plt.show() # Display posterior distributions

plt.close()
```

低值试样四参数拟合结果如图 B.1 所示, 试样分析结果 x = (0.373 ± 0.0210) ng/mL, 即 x 值的不确定 $u(x)$ = 0.0210 ng/mL, 取扩展因子 k=2, 则扩展不确定度 U=2×0.021=0.042 ng/mL, 试样分析结果可以表示为 (0.373 ± 0.042) ng/mL。



图B.1 低值试样四参数拟合计算结果

如果使用试剂盒自带的标准溶液直接测定，这些标准溶液没有提供不确定度也无法通过分析确定不确定度，则忽略掉标准溶液的不确定度，将 $u(x_i)$ 均设置为 10^{-6} ng/mL，此时低值试样四参数拟合结果如图 B.2 所示，试样分析结果 x=（0.369 ± 0.017）ng/mL，即 x 值的不确定 $u(x)$= 0.017 ng/mL，取扩展因子 k=2，则扩展不确定度 U=2×0.017=0.034 ng/mL，试样分析结果可以表示为（0.369 ± 0.034）ng/mL。



图B.2 低值试样四参数拟合计算结果[$u(x_i)$ =0]

高值试样四参数拟合结果如图 B.3 所示，试样分析结果 x =（1.164 ± 0.095）ng/mL，即 x 值的不确定 $u(x)$ = 0.095 ng/mL，取扩展因子 k=2，则扩展不确定度 U=2×0.095 = 0.19 ng/mL，试样分析结果可以表示为（1.16 ± 0.19）ng/mL。

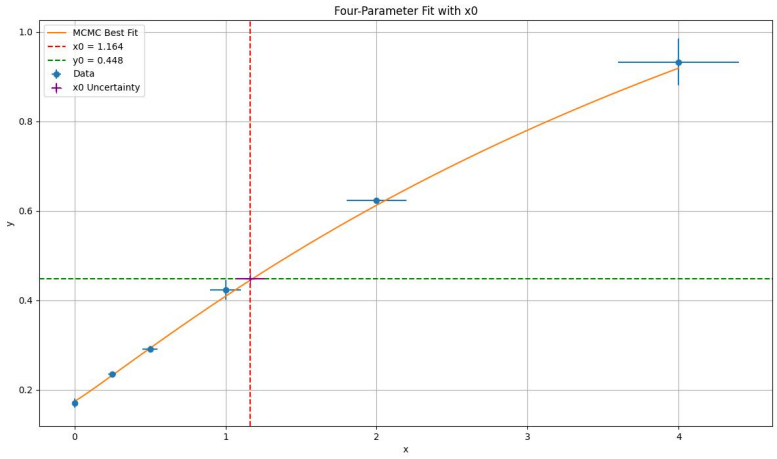
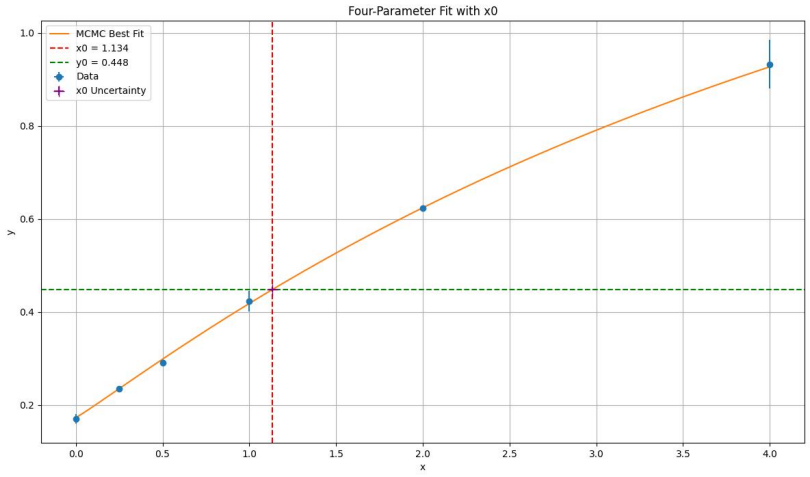


图 B.3 高值试样四参数拟合计算结果

如果使用试剂盒自带的标准溶液直接测定，这些标准溶液没有提供不确定度也无法通过分析确定不确定度，则忽略掉标准溶液的不确定度，将 $u(x_i)$ 均设置为 10^{-6} ng/mL，此时高值试样四参数拟合结果如图 B.4 所示，试样分析结果 x=（1.134 ± 0.020）ng/mL，即 x 值的不确定 $u(x)$ =0.020 ng/mL，取扩展因子 k=2，则扩展不确定度 U= 2×0.020 = 0.040 ng/mL，试样分析结果可以表示为（1.134 ± 0.040）ng/mL。



图B.4高值试样四参数拟合计算结果[$u(x_i)$ =0]

附录 C　三次样条插值拟合免疫分析结果不确定度的评定示例

（资料性）

C.1、实验方法

同 A.1。

C.2、实验结果

同 A.2。

C.3、$u(x_i)$, $u(y_i)$和 $u(y)$的计算

同 A.3。

C.4、$u(x)$和 U 的计算

通过Python编程计算u（x），代码如下：

```
import matplotlib

matplotlib.use('TkAgg')

import numpy as np

import matplotlib.pyplot as plt

from scipy.interpolate import CubicSpline

from scipy.optimize import brentq

import emcee

import corner

import seaborn as sns


# 设置 seaborn 样式

sns.set(style="whitegrid")
```

```
    # 定义数据
    x_data = np.array([0, 0.25, 0.50, 1.00, 2.00, 4.00])
    y_data = np.array([0.17, 0.235, 0.2905, 0.423, 0.6235, 0.9325])
    u_x_data = np.array([1e-6, 0.02512, 0.05017, 0.1002, 0.2002, 0.4000])
    u_y_data = np.array([0.010029883, 1e-6, 0.006895545, 0.022567238, 0.000626868, 0.05203002])
    y0 = 0.2650
    u_y0 = 0.004020

    n_params = 13    # 6 x_real, 6 y_real, 1 y0_true

    def log_prior(theta):
        x_real = theta[:6]
        y_real = theta[6:12]
        y0_true = theta[12]

        # 检查x_real是否递增
        if not np.all(np.diff(x_real) > 0):
            return -np.inf

        # 计算先验概率
        log_p = 0.0
        # x_real的先验
        log_p += np.sum(-0.5 * ((x_real - x_data) ** 2 / (u_x_data ** 2)) - np.log(u_x_data * np.sqrt(2 *
np.pi)))
        # y_real的先验
```

```
        log_p += np.sum(-0.5 * ((y_real - y_data) ** 2 / (u_y_data ** 2)) - np.log(u_y_data * np.sqrt(2 *
np.pi)))
        # y0_true的先验
        log_p += -0.5 * ((y0_true - y0) ** 2 / (u_y0 ** 2)) - np.log(u_y0 * np.sqrt(2 * np.pi))

        return log_p

    def log_probability(theta):
        lp = log_prior(theta)
        if not np.isfinite(lp):
            return -np.inf

        x_real = theta[:6]
        y_real = theta[6:12]
        y0_true = theta[12]

        # 构造三次样条插值
        try:
            cs = CubicSpline(x_real, y_real, extrapolate=False)
        except:
            return -np.inf

        # 定义求解x0的函数
        def func(x):
            return cs(x) - y0_true
```

```
        x_min = x_real.min()
        x_max = x_real.max()

        # 检查端点处的符号
        try:
            f_min = func(x_min)
            f_max = func(x_max)
        except:
            return -np.inf

        if np.sign(f_min) == np.sign(f_max):
            return -np.inf  # 没有根

        # 寻找根
        try:
            x0_val = brentq(func, x_min, x_max)
        except:
            return -np.inf

        return lp

# 生成初始位置
n_walkers = 50

def generate_initial_positions(n_walkers):
        initial = np.zeros((n_walkers, n_params))
```

```
    for i in range(n_walkers):
        valid = False
        while not valid:
            x_real = x_data + np.random.normal(0, u_x_data / 10)
            if np.all(np.diff(x_real) > 0):
                valid = True
        y_real = y_data + np.random.normal(0, u_y_data / 10)
        y0_true = y0 + np.random.normal(0, u_y0 / 10)
        initial[i] = np.concatenate([x_real, y_real, [y0_true]])
    return initial

initial_positions = generate_initial_positions(n_walkers)

# 运行MCMC
sampler = emcee.EnsembleSampler(n_walkers, n_params, log_probability)

# Burn-in
print("Running burn-in...")
n_burn = 1000
state = sampler.run_mcmc(initial_positions, n_burn, progress=True)
sampler.reset()

# 主采样
n_steps = 5000
print("Running production...")
sampler.run_mcmc(state, n_steps, progress=True)
```

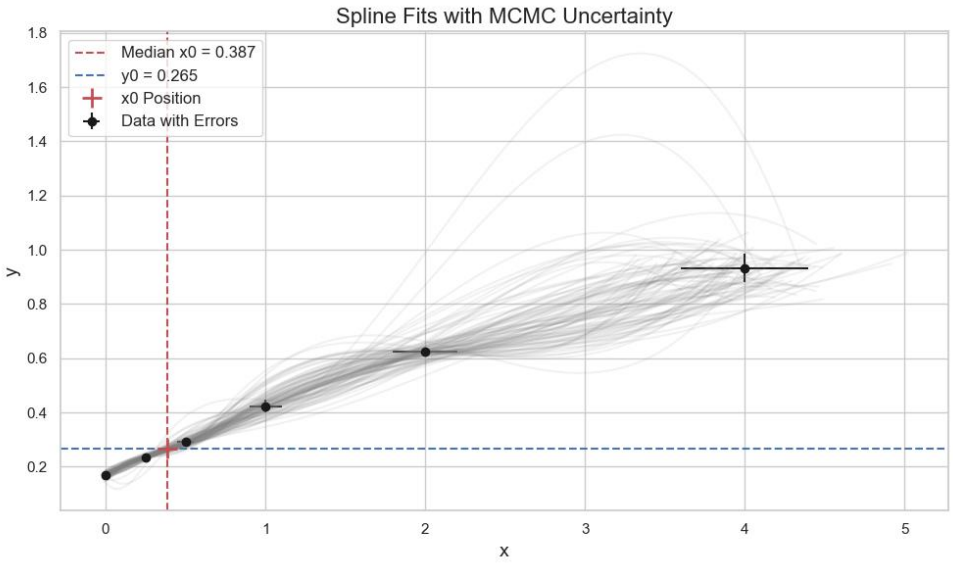
```
# 获取样本
samples = sampler.get_chain(flat=True)

# 计算x0的后验样本
x0_samples = []
for theta in samples:
    x_real = theta[:6]
    y_real = theta[6:12]
    y0_true = theta[12]

    try:
        cs = CubicSpline(x_real, y_real, extrapolate=False)
    except:
        continue

    def func(x):
        return cs(x) - y0_true

    x_min = x_real.min()
    x_max = x_real.max()

    try:
        f_min = func(x_min)
        f_max = func(x_max)
    except:
```

```
            continue

        if np.sign(f_min) == np.sign(f_max):
            continue

        try:
            x0 = brentq(func, x_min, x_max)
            x0_samples.append(x0)
        except:
            continue

x0_samples = np.array(x0_samples)

# 检查x0_samples是否为空
if len(x0_samples) == 0:
    raise ValueError("No valid x0 samples found.")

# 计算x0的中位数和不确定度
x0_median = np.median(x0_samples)
ux0 = np.std(x0_samples)

# 输出x0的值
print(f"Median x0: {x0_median:.4f}")
print(f"Estimated uncertainty u_x0: {ux0:.4f}")

# 可视化x0的后验分布
```

```
plt.figure(figsize=(10, 6))
plt.hist(x0_samples, bins=50, density=True, alpha=0.6, color='blue')
plt.title(f'Posterior Distribution of x0\n $u_{{x0}}$ = {ux0:.4f}', fontsize=16)
plt.xlabel('x0', fontsize=14)
plt.ylabel('Probability Density', fontsize=14)
plt.show()

# 绘制拟合曲线
plt.figure(figsize=(10, 6))
plt.errorbar(x_data, y_data, xerr=u_x_data, yerr=u_y_data, fmt='o', color='k', label='Data with Errors')

# 绘制部分后验样本的曲线
for i in np.random.choice(len(samples), 100):
    theta = samples[i]
    x_real = theta[:6]
    y_real = theta[6:12]
    try:
        # 构造三次样条插值
        cs = CubicSpline(x_real, y_real, extrapolate=False)
        x_plot = np.linspace(x_real.min(), x_real.max(), 100)
        y_plot = cs(x_plot)
        plt.plot(x_plot, y_plot, color='gray', alpha=0.1, label='Posterior Samples' if i == 0 else None)
    except:
        continue  # 如果插值失败，跳过该样本

# 绘制中位数的x0
```

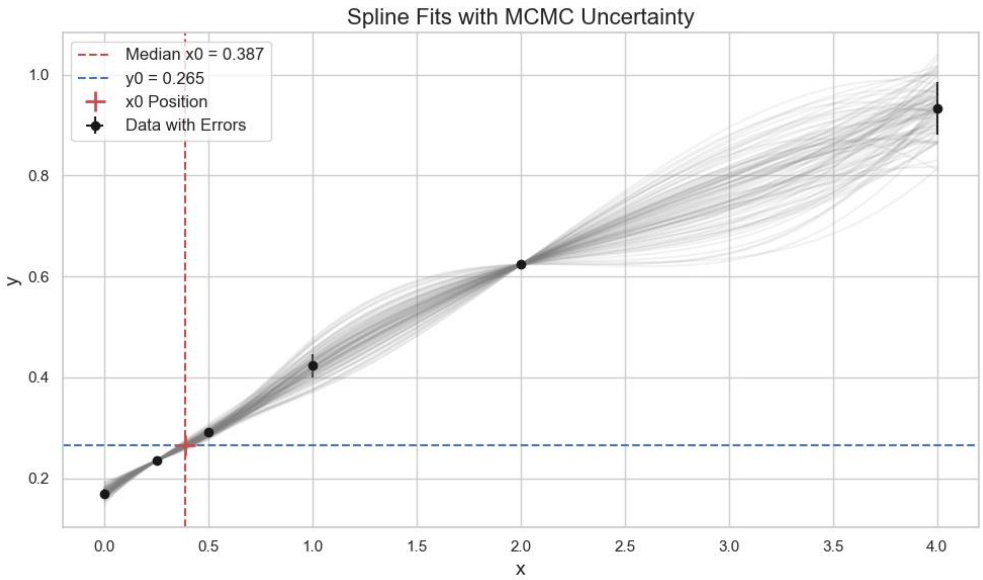
```
plt.axvline(x0_median, color='r', linestyle='--', label=f'Median x0 = {x0_median:.3f}')
plt.axhline(y0, color='b', linestyle='--', label=f'y0 = {y0}')
plt.plot(x0_median, y0, 'r+', markersize=15, markeredgewidth=2, label='x0 Position')

# 添加标题和标签
plt.xlabel('x', fontsize=14)
plt.ylabel('y', fontsize=14)
plt.title('Spline Fits with MCMC Uncertainty', fontsize=16)

# 添加图例
plt.legend(loc='upper left', fontsize=12)

# 调整布局
plt.tight_layout()

# 显示图形
plt.show()

# 使用corner绘制参数的后验分布
corner_labels = [f'x_real_{i+1}' for i in range(6)] + [f'y_real_{i+1}' for i in range(6)] + ['y0_true']
corner.corner(samples, labels=corner_labels, truths=np.median(samples, axis=0))
plt.show()
```

低值试样三次样条插值拟合结果如图 C.1 所示，试样分析结果 x =（0.387 ± 0.056）ng/mL，即 x 值的不确定 $u(x)$ = 0.056 ng/mL，取扩展因子 k=2，则扩展不确定度 U=2×0.056=0.12 ng/mL，试样分析结果可以表示为（0.39 ± 0.12) ng/mL。



图C.1 低值试样三次样条插值拟合计算结果

如果使用试剂盒自带的标准溶液直接测定，这些标准溶液没有提供不确定度也无法通过分析确定不确定度，则忽略掉标准溶液的不确定度，将 $u(x_i)$ 均设置为 10^{-6} ng/mL，此时低值试样三次样条插值拟合结果如图 C.2 所示，试样分析结果 x= (0.387 ± 0.028) ng/mL，即 x 值的不确定 $u(x)$= 0.028 ng/mL，取扩展因子 k=2，则扩展不确定度 U=2×0.028 = 0.056 ng/mL，试样分析结果可以表示为 (0.387 ± 0.056) ng/mL。



图C.2 低值试样三次样条插值拟合计算结果[$u(x_i)$ =0]

高值试样三次样条插值拟合结果如图 C.3 所示，试样分析结果 x =（1.09 ± 0.15）ng/mL，即 x 值的不确定 $u(x)$ = 0.15 ng/mL，取扩展因子 k=2，则扩展不确定度 U=2×0.15=0.30 ng/mL，试样分析结果可以表示为（1.09 ± 0.30）ng/mL。

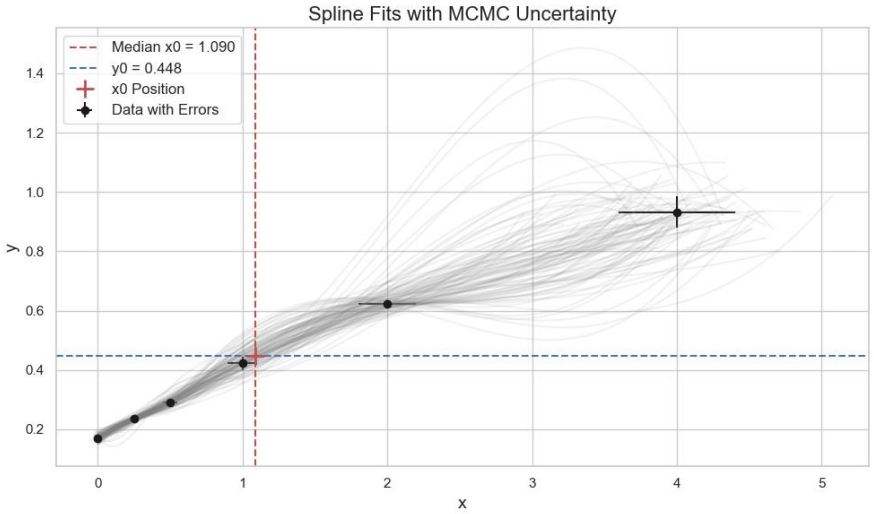


图 C.3 高值试样三次样条插值拟合计算结果

如果使用试剂盒自带的标准溶液直接测定，这些标准溶液没有提供不确定度也无法通过分析确定不确定度，则忽略掉标准溶液的不确定度，将 $u(x_i)$ 均设置为 10^{-6} ng/mL，此时高值试样三次样条插值拟合结果如图 C.4 所示，试样分析结果 x =（1.10 ± 0.11）ng/mL，即 x 值的不确定 $u(x)$= 0.11 ng/mL，取扩展因子 k=2, 则扩展不确定度 U=2×0.11= 0.22 ng/mL, 试样分析结果可以表示为 (1.10 ± 0.22) ng/mL。

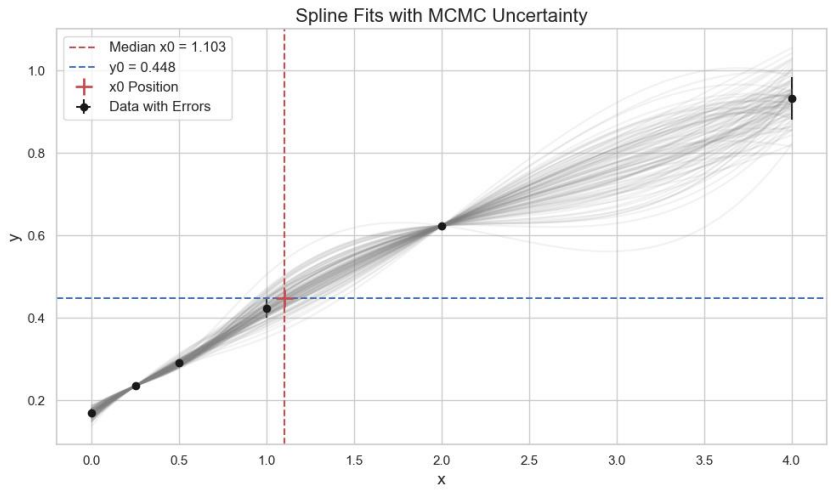


图 C.4 高值试样三次样条插值拟合计算结果[$u(x_i)$ =0]