課題

1. 厚さ 100 nmの a-SiO_2 の単位面積当たり静電容量 C_{OX} を求めよ。
   a-SiO_2 の 比誘電率は $\varepsilon_r = 11.9$ とする。

2. TransferCurve.xlsxのデータから、飽和移動度を求めよ
   電極幅 W = 300 μm, L = 50 μm とする。
   飽和移動度を求める際の Vg、Vd は各自で選ぶこと。
   その値を選んだ理由も説明せよ。

PowerPoint 等 のプレゼンテーションファイルにして提出
期限：今日の17：00までに
できたところまでで可

最尤推定法

尤度関数とは:
事象 (x_i) *が起こる確率を、既知のパラメータ* (a_k) *の確率密度関数*

$$P(X = x_i|a_k) = \prod_i \frac{1}{\sqrt{2\pi\sigma_i}} exp\left[-\frac{\varepsilon_i(a_k)^2}{2{\sigma_i}^2}\right]$$

などとすると、逆に $X = (x_i)$ *がわかっているとし、*
パラメータ a_k がどれだけ尤もらしいか (尤度) を表す確率密度関数とみなし、
上記の確率密度関数を変数 (a_i) の関数として
尤度関数 $P(a_i) = P(x_i|a_i)$という。

最尤推定法

誤差 $\varepsilon_i = f(x_i, a_i) - y_i$ が分散 σ_i の正規分布に従うとする。データ(x_i, y_i)に対するパラメータ (a_i) の尤度関数は

$$\boldsymbol{P(a_i) = \prod_i \frac{1}{\sqrt{2\pi\sigma_i}} \exp\left[-\frac{{\varepsilon_i}^2}{2{\sigma_i}^2}\right]}$$

尤度を最大化するパラメータを求めるのが「最尤推定法」。

$$\boldsymbol{max\ P(a_i) = max \ln P(a_i) = min \sum_i \frac{{\varepsilon_i}^2}{{\sigma_i}^2}}$$

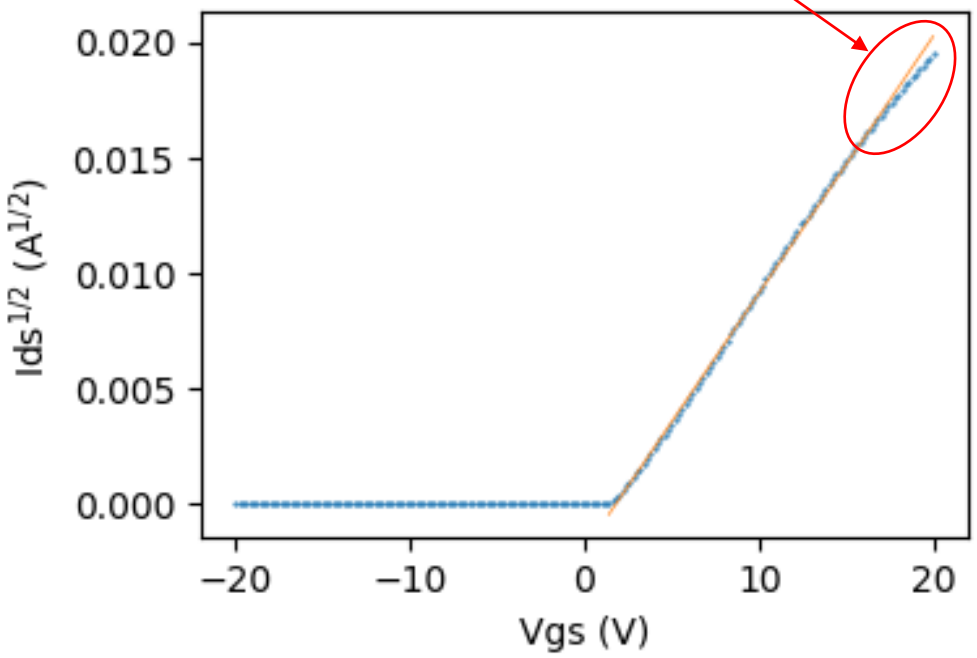
: 最小二乗法に一致する

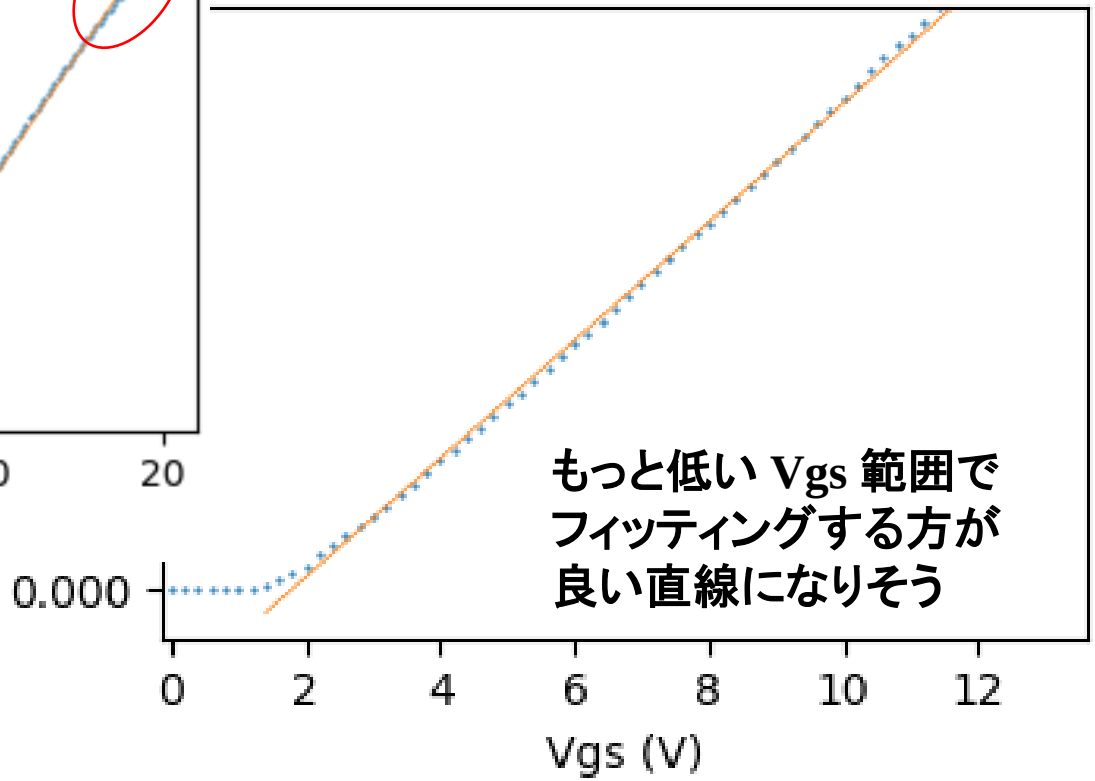
フィッティング範囲の妥当性

TFTiv_errorcheck.py

フィッティング範囲 2.0 ~ 10.0V

高 Vgs では Vds < Vp = Vgs – Vth となり、
直線から外れる
最小二乗に入れてはいけない





線形最小二乗法 $y = a + bx$ の誤差

酒井英行訳、M.C.Barford著、実験精度と誤差、丸善

$f(x_i) = a + bx_i$

$\varepsilon_i = f(x_i) - (a + bx_i)$　　目的関数 $S = \sum {\varepsilon_i}^2$ を最小化

$$\begin{pmatrix} n & s_x \\ s_x & s_{xx} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} s_y \\ s_{xy} \end{pmatrix} \quad s_x = \sum x_i, s_y = \sum y_i,\ s_{xx} = \sum {x_i}^2,\ \ s_{xy} = \sum x_i y_i$$

$$\begin{pmatrix} a \\ b \end{pmatrix} = \frac{1}{\Delta} \begin{pmatrix} s_{xx} & -s_x \\ -s_x & n \end{pmatrix} \begin{pmatrix} s_y \\ s_{xy} \end{pmatrix} \qquad \Delta = n s_{xx} - {s_x}^2 = n \sum (x_i - \langle x \rangle)^2$$

$$b = \frac{n s_{xy} - s_x s_y}{\Delta} = \frac{\sum (x_i - \langle x \rangle)(y_i - \langle y \rangle)}{\sum (x_i - \langle x \rangle)^2} \qquad a = \frac{s_{xx} s_y - s_x s_{xy}}{\Delta} = \langle y \rangle - b \langle x \rangle$$

$${\sigma_y}^2 = \langle y^2 \rangle - \langle y \rangle^2 - \frac{(\langle xy \rangle - \langle x \rangle \langle y \rangle)^2}{\langle x^2 \rangle - \langle x \rangle^2} = \frac{1}{n^2} \left[n s_{yy} - {s_y}^2 - \frac{\left(n s_{xy} - s_x s_y \right)^2}{n s_{xx} - {s_x}^2} \right]$$

標準誤差: $S_a = \dfrac{\sigma_y \sqrt{\langle x^2 \rangle}}{\sqrt{(n-2)\left(\langle x^2 \rangle - \langle x \rangle^2\right)}}$　　$S_b = \dfrac{\sigma_y}{\sqrt{(n-2)\left(\langle x^2 \rangle - \langle x \rangle^2\right)}}$

相関係数: $r = s_{xy} / \sqrt{s_{xx} s_{yy}}$

誤差の計算

誤差の伝播則

変数 a, b の標準誤差σ_a, σ_b が既知の場合、$f(a,b)$ の誤差は

$$\delta f(a,b) = \left(\frac{\partial f}{\partial a}\right)_b \delta a + \left(\frac{\partial f}{\partial b}\right)_a \delta b$$

$\delta a, \delta b$ **が正規分布に従う場合、**$\delta f(a,b)$**の標準偏差**σ_f**は**

$$\sigma_f = \sqrt{\left[\left(\frac{\partial f}{\partial a}\right)_b \sigma_a\right]^2 + \left[\left(\frac{\partial f}{\partial b}\right)_a \sigma_b\right]^2}$$

最小自乗法で y = a + bx の標準誤差が得られたら・・・

・ $V_{th}(a,b) = -a/b$

$$\delta V_{th} = -\delta a/b + a\delta b/b^2$$

$$\boldsymbol{\sigma_{Vth} = \sqrt{\left[\left(\frac{1}{b}\right)\sigma_a\right]^2 + \left[\left(\frac{a}{b^2}\right)\sigma_b\right]^2}}$$

$$\delta \log V_{th} = \frac{\delta V_{th}}{V_{th}} = \delta a/a - \delta b/b$$

$$\boldsymbol{\sigma_{Vth} = V_{th}\sqrt{\left[\left(\frac{1}{a}\right)\sigma_a\right]^2 + \left[\left(\frac{1}{b}\right)\sigma_b\right]^2}}$$

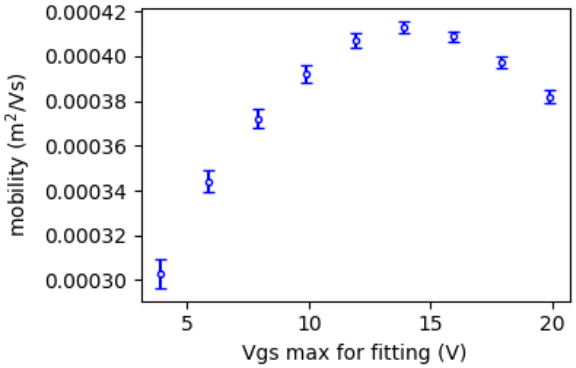
・ $\mu(a,b) = b^2/\sqrt{\frac{C_{ox}}{2L}}$

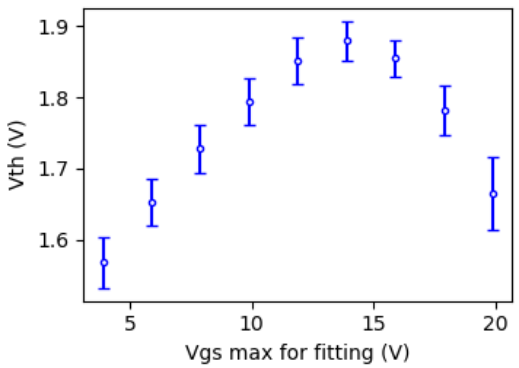
$$\delta \log \mu = \frac{\delta \mu}{\mu} = \delta[const + 2\ln b] = \frac{\delta b}{b}$$

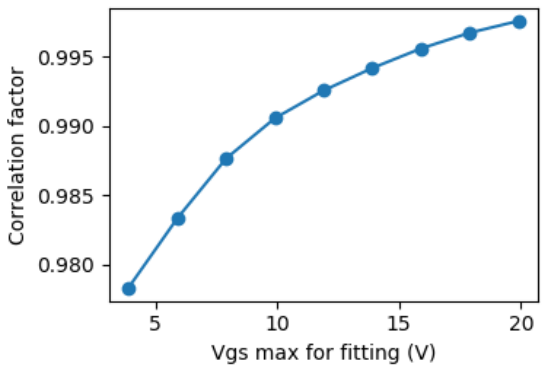
$$\boldsymbol{\sigma_\mu = 2\mu\sigma_b/b}$$

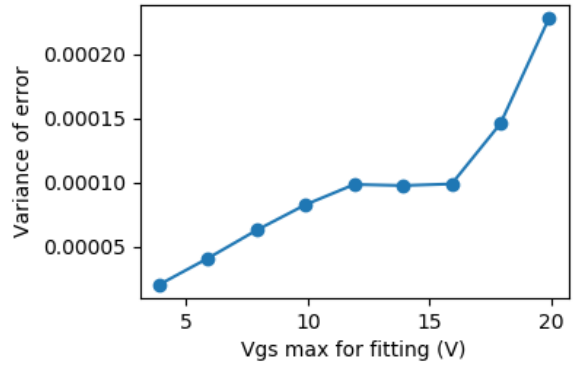
最小二乗法の誤差と相関係数

TFTiv_errorcheck.py









- フィッティング範囲を広げると**相関係数**は 1.0 に近づく
  x範囲が広いと相関係数は大きくなりやすい。
  必ずしも良い直線範囲の指標にはなっていない
- **誤差ε_iの標準偏差 σ_{ε_i}** は Vgs > 16V で明確に大きくなる:
  Vgs > 16 Vのデータを入れてはいけない
- 移動度、Vth は14Vまでのデータを入れると最大値をとる
  意味はない。フィッティング範囲の妥当性の検証が重要
- **標準誤差** は **1σ** で表示していることに注意。
  3σ を見込むのが普通

プログラム (抜粋)

TFTiv_errorcheck.py

```
# 誤差、相関係数計算付き 一次多項式線形最小二乗
def lsq1(x, y, iPrint = 0):
    n = len(x)
# 統計量の計算
# si: 変数 i の和                avi: i の平均
# sij: 変数 i と j の積の和      sij: i*jの平均
    sx = sum(x)
    avx = sx / n
    sy = sum(y)
    avy = sy / n
    sxx = sum([x[i] * x[i] for i in range(n)])
    avxx = sxx /n
    sxy = sum([x[i] * y[i] for i in range(n)])
    avxy = sxy /n
    syy = sum([y[i] * y[i] for i in range(n)])
    avyy = syy /n
    delta = n * sxx - sx * sx

# y = a + bx
    b = (n * sxy - sx * sy) / delta
    a = avy - b * avx

# 残差の二乗和 ∑εi^2、誤差の標準偏差 σ(εi)
    sum_ei2 = sum([pow(y[i] - a - b * x[i], 2) for i in range(n)])
    sigma_ei = sqrt(sum_ei2 / (n - 1))
    sigma_y2 = avyy - avy * avy - pow(avxy - avx * avy, 2) / (avxx -
avx * avx)
    sigma_y = sqrt(sigma_y2)

# パラメータ a, b の標準誤差と相関係数
Sa = sigma_y * sqrt(avxx) / sqrt((n - 2) * (avxx - avx * avx))
    Sb = sigma_y / sqrt((n - 2) * (avxx - avx * avx))
    r  = sxy / sqrt(sxx * syy)
```

```
# 戻り値が多いので、統計量、標準誤差、相関係数は
# 辞書変数 (ハッシュ、連想配列) で返す
# 辞書変数の値は、{key:val}
res = {'sx': sx, 'sy': sy, 'sxx': sxx, 'sxy': sxy, 'syy': syy,
        'Sa': Sa, 'Sb': Sb, 'r': r, 'sigma_ei': sigma_ei, 'residual': sum_ei2}
    return a, b, res

def main():
# 最小二乗を実行
    ai = lsq1(xfit, yfit, 0)
# 辞書変数は ai[2] に入る
    res = ai[2]
# a, b の標準誤差、相関係数
# 辞書変数の要素 は 変数名[key]で受け取る
    Sa = res['Sa']
    Sb = res['Sb']
    r  = res['r']

    Vth  = -ai[0] / ai[1]
    grad = ai[1]
    mu = grad * grad / (W * Cox / 2.0 / L)

# 誤差伝播則からVth と μ の標準誤差を計算
    SVth = sqrt(pow(Sa / ai[1], 2) + pow(Sb * ai[0] / ai[1] / ai[1], 2))
    Smu = 2.0 * mu * Sb / ai[1]

# エラーバー付きグラフのプロット
  ax4.errorbar(xecheck, ymu, yerr = ySmu,
          capsize = 3.0, fmt = ‘o’, markersize = 3.0, ecolor = ‘b’,
           markeredgecolor = 'b', color = 'w')
```