```
begin
    import Pkg
    Pkg.activate(".")
    using CSV, DataFrames, EpitaxialDeposition, LsqFit
end
```

df =

	y	r
1	0.0	0.0
2	0.0297645	2.56617
3	0.0451678	3.64415
4	0.110372	3.99732
5	0.150266	3.56928
6	0.250855	1.88385
7	0.305378	1.3645
8	0.370323	0.0

```
df = CSV.read("Steinmaier_1423K.csv", DataFrame)
```

```
begin
    P = 100.
    T = 1423.
    t_max = 3600.

    gaseous_species = Symbol.(["SiCl₄(t)", "SiCl₂(t)", "H₂(t)", "HCl(t)"])
end;
```

```
function model(y, p)
    # set model parameters
```

```
    PARAMS[:kKp][:k] = p[1]
    PARAMS[:kKp][:a] = p[2]
    PARAMS[:kKp][:b] = p[3]

    # solve ODE system and find film growth rates
    sol = [run_simulation(y, P, T, t_max) for y in y]
    δ = [film_thickness.(sol[:, "Si_dep(t)"], sol[:, "Si_etch(t)"]) for sol in sol]
    dδ = [estimate_derivative(δ) for δ in δ]

    # calculate mole fractions
    all_gases =
        [map(r -> sum([r[x] for x in gaseous_species]), eachrow(sol)) for sol in sol]
    molfrac_SiCl4 =
        [sol[:, gaseous_species[1]] ./ all_gases[i] for (i, sol) in enumerate(sol)]
    # find index of timepoint w/ mole fraction ~y
    idx =
        [argmin(abs.(molfrac_SiCl4 .- y[i])) for (i, molfrac_SiCl4) in
        enumerate(molfrac_SiCl4)]
    # return the film growth rate for the given mole fraction
    return [dδ[i][idx] for (i, idx) in enumerate(idx)]
end;
```

```
fit =
  LsqFitResult([-6.03973e5, 10.38, 21158.1], [0.0, -0.886333, -2.44143, -2.18182, -1.4912, 0
```

```
fit = curve_fit(model, df.y, df.r,
    [
        PARAMS[:kKp][:k],
        PARAMS[:kKp][:Ea],
        PARAMS[:kKp][:b]
    ]
)
```

```
[-6.03973e5, 10.38, 21158.1]
```

```
fit.param
```