Supplementary Material

# Supplementary text

The supplementary mainly illustrates two numerical methods and equations used in the concentration calculation and optimization. Text S1 shows the differential equations of Eq 1 to Eq 7. Text S2 depicts the details of Quasi-Newton optimization.

**Text S1 Differential equations of concentration calculation**

Equations of ODE:

Period I (0 to *t*sa)

Period II (*t*sa to *t*p)

*Henry adsorption equation:*

*Langmuir adsorption equation:*

Period III (*t*p to *t*r1)

*Henry adsorption equation:*

*Langmuir adsorption equation:*

Period IV

Sub-period 1 (*t*r1 to *t*r2)

Sub-period 2 (*t*r2 to *t*s)

*Henry adsorption equation:*

*Langmuir adsorption equation:*

**Text S2 Quasi-Newton optimization**

For this problem, the root means square deviation between numerical results and experimental results is set to be the objective function, and the parameters *γ*, *α*, *h*mix, , *B*, *C* are needed to be optimized.

where is numerical results, and is the experimental result.

In the Langmuir Adsorption method, ; in the Henry Adsorption method, .

The numerical solution of *f(x)* is calculated by the *ODE* method in Text S1, on the other hand, differential equations are applied on each parameter to form Jacob matrix to derive the

The same theory in the Langmuir adsorption equation:

Using the numerical solution of and , the iteration is conducted to do the optimization of the parameters in the models, such as *γ, α*, *h*mix, *B*, and *C* in the Langmuir model and *γ, α*, *h*mix and *k*d in the Henry model.

Next, in the Quasi-Newton method, the algorithm chooses a direction *dk* and searches along this direction from the current iterate *xk* for a new iterate with a lower function value. The distance to move along *dk* can be found by approximately solving the following one-dimensional minimization problem to find a step length *a*:

where the step length *ak* is chosen to satisfy the Wolfe conditions, *a0* is usually 1 in the Quasi-Newton method:

where , .

The minimizer *dk* of this convex quadratic model, which we can write explicitly as:

*Bk* is also computed afresh at every iteration, and we use BFGS updating formula to determine the value of *Bk*:

where , .

Then do it again in iteration until .