appVersion(4) = "1.0.8348.30405"

Notes Use Φ(x,β), now the function could be any other

Usually, don't redefine variables from the args in the procedure body. (like β)

```
LMSA(Φ(2), x, y, βg, Tol) :=
    [ Eps := Tol  eta1 := √Eps  eta2 := eta1  h := eta1  stop := 0 ]
    [ β := βg  m := rows(x)  n := rows(β) ]
    fx(β) :=
        for i ∈ [1..m]
            ff_i := eval(Φ(x_i, β) − y_i)
        ff
    derfh(β) :=
        for i ∈ [1..m]
            for j ∈ [1..n]
                β_j := β_j + h
                jjh1 := fx(β)_i
                β_j := β_j − 2·h
                jjh2 := fx(β)_i
                jjh0_(i,j) := (jjh1 − jjh2)/(2·h)
                β_j := β_j + h
        jjh0
    [ A := Zeros(n, n)  g := Zeros(n)  J := Zero(m, n) ]
    [ Fn := Zero(1)  dL := Zero(1)  dF := Zero(1)  F := Zero(1) ]
    [ f := fx(β)  J := derfh(β) ]
    [ A := J^T · J  g := J^T · f  ng := normi(g) ]
    [ F := (f^T · f)/2  mu := eta1 · max(Diag(A))  nu := 2 ]
    while ¬stop
        if ng ≤ eta2
            stop := 1
        else
            try
                p := invert(A + mu · identity(n)) · (−g)
            on error
                break
            np := normi(β)
            nx := eta2 + normi(β)
            if np ≤ eta2 · nx
                stop := 2
            else
                if ¬stop
                    xnew := β + p
                    fn := fx(xnew)
                    Jn := derfh(xnew)
                    Fn := (fn^T · fn)/2
                    dL := (p^T · (mu · p − g))/2
```

```
β := xnew
F := Fn
J := Jn
f := fn
A := J^T · J
g := J^T · f
ng := normi(g)
mu := eval(mu · Max(1/3, 1 − (2 · dF_1/dL_1 − 1)^3))
nu := 2
else
  mu := mu · nu
  nu := 2 · nu
else
  ""
β
```

$$M := \begin{bmatrix} \text{"x,time,min"} & \text{"y,dye concentration, g/gal"} \\ 3 & 0 \\ 5 & 0 \\ 7 & 0 \\ 11 & 0 \\ 14 & 1 \\ 16 & 3 \\ 19 & 13 \\ 21 & 22 \\ 24 & 29 \\ 25 & 26 \\ 26 & 18 \\ 27 & 8 \\ 29 & 3 \\ 33 & 0 \\ 36 & 0 \\ 40 & 0 \\ 44 & 0 \end{bmatrix}$$

Keep lowercase x undefined, if you want to use SMath's plots.

$X := \text{submatrix}(M, 5, 15, 1, 1)$

$Y := \text{submatrix}(M, 5, 15, 2, 2)$

$$F(x, \beta) := \beta_1 \cdot \exp\left(-\frac{(x - \beta_2)^2}{2 \cdot (\beta_3)^2}\right)$$

very low tol

$Tol := 10^{-12}$

guess value

$$\beta_0 := \begin{bmatrix} 10 \\ 10 \\ 10 \end{bmatrix}$$

$t0 := \text{time}(1)$

$\beta_1 := LMSA(F(\beta\#, x\#), X, Y, \beta_0, Tol) = \begin{bmatrix} 30.9827 \\ 22.9871 \\ 2.7954 \end{bmatrix}$

$\text{time}(1) - t0 = 8.845\ s$

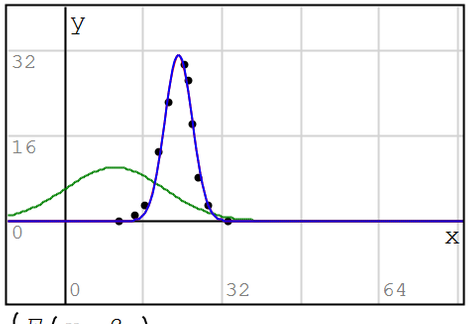
$$fit(\beta) := \sum_{k=1}^{\text{length}(X)} \left(F(X_k, \beta) - Y_k\right)^2$$

$t0 := \text{time}(1)$

$\beta_2 := \text{al\_nleqsolve}(\beta_0, fit) = \begin{bmatrix} 30.8152 \\ 22.9864 \\ 2.8078 \end{bmatrix}$

using the same guess for both algorithms.

$\text{time}(1) - t0 = 3.607\ s$



$F(x, \beta_1)$
$F(x, \beta_2)$
$F(x, \beta_0)$
$\text{augment}(X, Y, ".")$

$fit(\beta_1) = 25.0271$

$fit(\beta_2) = 25.0639$

$fit(\beta_0) = 1929.9695$

Your algorithm is "better" than the nle, at least for it with the default values ... well, ok, only at the 4th significant figure and with a very small Tol.

Plots with β1 and β2 are essentially the same.

also plot your first approx,
with the guess value, just in case.