

Dérivée à plusieurs variables

De droite tangente à plan tangent

La dérivée univariée au point \mathbf{a} correspond à la pente de la **droite tangente** à la fonction en \mathbf{a} . Pour une fonction bivariée, la dérivée dans une direction \mathbf{d} correspond à la pente du **plan tangent** à la fonction en \mathbf{a} dans la direction \mathbf{d} .

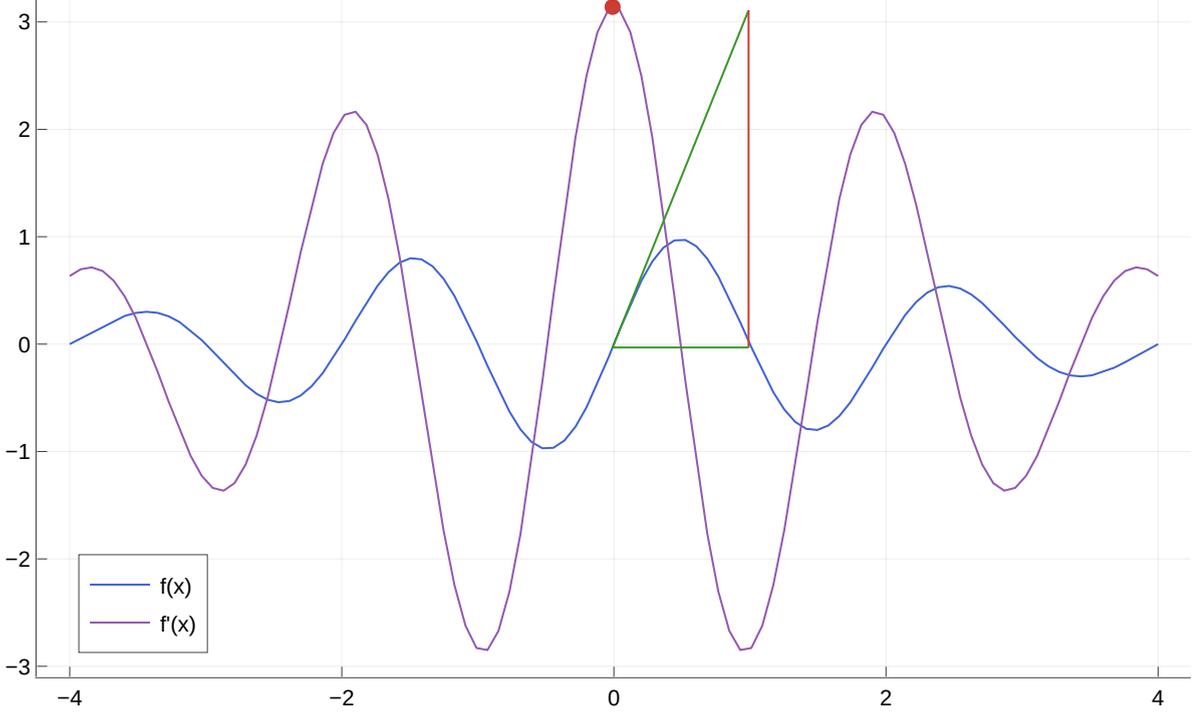
► PlotlyBackend()

1 `plotly()`



```
Precompiling PlotlyBase...
 3682.1 ms ✓ PlotlyBase
 1 dependency successfully precompiled in 4 seconds. 47 already precompiled.
Precompiling PlotlyKaleido...
 462.9 ms ✓ Kaleido_jll
 601.7 ms ✓ PlotlyKaleido
 2 dependencies successfully precompiled in 1 seconds. 29 already precompiled.
```



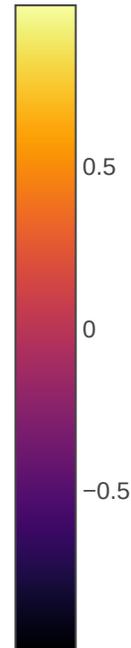


f(x)
f'(x)

```
1 tangent_line(x -> sin(x * pi) * exp(-x^2 / 10), x0, x1, cx)
```

-0.01

-0.01



```
1 tangent_plane((x, y) -> sin(x * pi) * exp(-y^2 / 4), x0, x1, y0, y1, cx, cy)
```

Dérivée directionnelle ⇔

Que vaut la dérivée dans la direction rouge $(1, 1)$ si on connaît les dérivées vertes en x : $\partial f / \partial x$ et en y : $\partial f / \partial y$? Un plan est défini par un point et deux vecteurs donc le plan bleu est entièrement défini par le point $(x, f(x))$ et les deux vecteurs $(1, 0, \partial f / \partial x)$ et $(0, 1, \partial f / \partial y)$. En utilisant la linéarité du plan, quelle est la valeur de ? pour que le vecteur $(1, 1, ?)$ soit dans le plan ?

$$(1, 1, \partial f / \partial x + \partial f / \partial y)$$

En général, pour une direction arbitraire, on a

$$(d_x, d_y, d_x \cdot \partial f / \partial x + d_y \cdot \partial f / \partial y)$$

La dérivée en direction (d_x, d_y) est donc obtenue par produit scalaire entre le vecteur (d_x, d_y) et le gradient $(\partial f / \partial x, \partial f / \partial y)$. Voici trois notations équivalentes pour écrire ce produit scalaire:

$$\langle (d_x, d_y), (\partial f / \partial x, \partial f / \partial y) \rangle = [d_x \quad d_y] \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix} = d_x \cdot \partial f / \partial x + d_y \cdot \partial f / \partial y$$

Le gradient est souvent dénoté avec le symbol ∇ :

$$\nabla f = (\partial f / \partial x, \partial f / \partial y)$$

Calculer le gradient à la main \Leftrightarrow

Comment calculer les valeurs de $\partial f / \partial x$ et $\partial f / \partial y$? Commençant par voir comment faire ça à la main. Nous verrons au cours suivant comment le calculer algorithmiquement. L'astuce: pour calculer $\frac{\partial}{\partial x} f(x, y)$, il faut voir y comme constant et donc voir f comme fonction de x uniquement. Par exemple:

$$\frac{\partial}{\partial x} \sin(x\pi) e^{-y^2/4} = \pi \cos(x\pi) e^{-y^2/4}$$

Idem pour y :

$$\frac{\partial}{\partial y} \sin(x\pi) e^{-y^2/4} = -\sin(x\pi) \frac{y}{2} e^{-y^2/4}$$

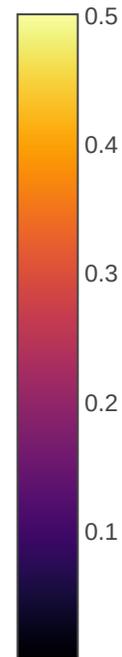
Quand les dérivées sont-elles dans un plan ? \Leftrightarrow

On dit dans ce cas que la fonction est *différentiable\$ en $(0, 0)$ et les dérivées directionelles sont donnée par produit scalaire avec le gradient. Quels sont les conditions pour la différentiabilité ?

Voyons un exemple:

$i_1 =$ 2

$i_2 =$ 2



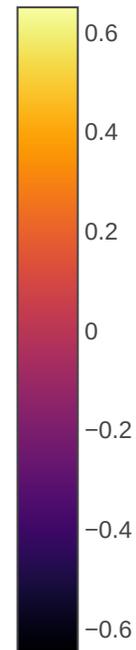
```
1 let
2   x = y = range(-1, stop = 1, length = 32) # nombre pair de points pour ne pas
   évaluer en (0, 0)
3   surface(x, y, (x, y) -> xi_1 * yi_2 / (xi_3 + yi_4))
4 end
```

$i_3 =$

$i_4 =$

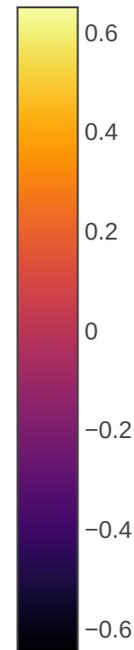
Pour quels valeurs de i_1, i_2, i_3, i_4 est ce que la fonction est continue en $(0, 0)$? Vous y répondez à la séance d'exercices! Ici, on se contentera de se fier au visuel.

Il est nécessaire que la fonction soit continue en $(0, 0)$, donc par exemple $xy/(x^2 + y^2)$ n'est pas continue. Y a-t-il des conditions sur $\partial f/\partial x$:



```
1 let
2   x = y = range(-1, stop = 1, length = 32)
3   function dfdx(x, y)
4     d = -xi_1 * yi_2 / (xi_3 + yi_4)2 * i_3 * x(i_3 - 1)
5     if i_1 > 0
6       d += i_1 * x(i_1 - 1) * yi_2 / (xi_3 + yi_4)
7     end
8     return d
9   end
10  surface(x, y, dfdx)
11 end
```

Et $\partial f / \partial y$:



```
1 let
2   x = y = range(-1, stop = 1, length = 32)
3   function dfdx(x, y)
4     d = -xi_1 * yi_2 / (xi_3 + yi_4)2 * i_4 * y(i_4 - 1)
5     if i_2 > 0
6       d += i_2 * xi_1 * y(i_2 - 1) / (xi_3 + yi_4)
7     end
8     return d
9   end
10  surface(x, y, dfdx)
11 end
```

Théorème: Si f est continue en a et que les dérivées partielles de f sont définies dans un voisinage de a et continues en a alors f est différentiable.

La Hessienne ⇔

Pour obtenir la dérivée second, on dérive à nouveau le gradient. On les notations:

$$\frac{\partial}{\partial x} \left(\frac{\partial}{\partial x} f \right) = \frac{\partial^2}{\partial x^2} f$$

$$\frac{\partial}{\partial x} \left(\frac{\partial}{\partial y} f \right) = \frac{\partial^2}{\partial xy} f$$

$$\frac{\partial}{\partial y} \left(\frac{\partial}{\partial x} f \right) = \frac{\partial^2}{\partial yx} f$$

$$\frac{\partial}{\partial y} \left(\frac{\partial}{\partial y} f \right) = \frac{\partial^2}{\partial y^2} f$$

Attention: ∂x^2 ne veut **pas** dire qu'on dérive par x^2 , c'est juste une notation pour dire qu'on dérive 2 fois par x .

Cette notation peut paraître mal choisie ! En effet, dans ∂xy et ∂yx , les parties xy et yx ressemblent à des produits. Comme le produit est commutatif, on a à tendance à penser que dériver par x puis y est égal à dériver par y puis x ... À moins que ça soit vraiment égal ?

Théorème Si f est différentiable alors $\frac{\partial^2}{\partial xy} f = \frac{\partial^2}{\partial yx} f$.

On aime mettre ces 4 valeurs dans une matrice nommée *Hessienne*. Par ce théorème, la matrice est donc symétrique!

$$\text{Hess}(f) = \begin{bmatrix} \frac{\partial^2}{\partial x^2} f & \frac{\partial^2}{\partial xy} f \\ \frac{\partial^2}{\partial xy} f & \frac{\partial^2}{\partial y^2} f \end{bmatrix}$$

Dérivée à plus de 2 variables \Leftrightarrow

Tous les résultats se généralisent sans broncher à plus de 2 variables. Considérons une fonction $f(x_1, x_2, \dots, x_n)$ à n variables. Le gradient est $\nabla f = (\partial f / \partial x_1, \partial f / \partial x_2, \dots, \partial f / \partial x_n)$ La dérivée dans la direction $\mathbf{d} = (d_1, d_2, \dots, d_n)$ est:

$$\langle \mathbf{d}, \nabla f \rangle = \mathbf{d}^T \nabla f = d_1 \cdot \partial f / \partial x_1 + \dots + d_n \cdot \partial f / \partial x_n$$

La Hessienne est toujours symétrique et devient:

$$\text{Hess}(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 x_2} & \frac{\partial^2 f}{\partial x_1 x_3} & \dots & \frac{\partial^2 f}{\partial x_1 x_n} \\ \frac{\partial^2 f}{\partial x_1 x_2} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 x_3} & \dots & \frac{\partial^2 f}{\partial x_2 x_n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 x_{n-1}} & \frac{\partial^2 f}{\partial x_2 x_{n-1}} & \dots & \frac{\partial^2 f}{\partial x_{n-1}^2} & \frac{\partial^2 f}{\partial x_{n-1} x_n} \\ \frac{\partial^2 f}{\partial x_1 x_n} & \frac{\partial^2 f}{\partial x_2 x_n} & \dots & \frac{\partial^2 f}{\partial x_{n-1} x_n} & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Utils ⇔

tangent_plane (generic function with 1 method)

```

1 function tangent_plane(f, x0, x1, y0, y1, cx, cy; Δ = 0.5, n = 32)
2     x = range(x0, stop = x1, length = n)
3     y = range(y0, stop = y1, length = n)
4     df(x, y) = ForwardDiff.gradient(xy -> f(xy...), [x, y])
5     surface(x, y, f, label = "f(x)", zlims = (minimum(f.(x, y')) + maximum(df.(x,
6     y'))), maximum(f.(x, y') + maximum(df.(x, y'))), legend=:bottomleft)
7     grad = df(cx, cy)
8     fc = f(cx, cy)
9     plane(x, y) = fc + grad' * [x - cx, y - cy]
10    plot!([cx, cx + 1], [cy, cy], [fc, fc + grad[1]], linewidth = 10, color =
11    Colors.JULIA_LOGO_COLORS.green, label = "")
12    plot!([cx, cx], [cy, cy + 1], [fc, fc + grad[2]], linewidth = 10, color =
13    Colors.JULIA_LOGO_COLORS.green, label = "")
14    return plot!([cx, cx+1], [cy, cy + 1], [fc, fc + grad[1] + grad[2]], linewidth
15    = 10, color = Colors.JULIA_LOGO_COLORS.red, label = "")
16    surface!( # works with plotly but not GR
17    range(cx - Δ, stop = cx + Δ, length = 3),
18    range(cy - Δ, stop = cy + Δ, length = 3),
19    plane,
20    color = [Colors.JULIA_LOGO_COLORS.blue],
21    label = "",
22    )
23 end

```

```
1 using LinearAlgebra, Plots, Colors, ForwardDiff, PlutoUI
```

```

Precompiling ForwardDiff...
3371.1 ms ✓ ForwardDiff
1 dependency successfully precompiled in 4 seconds. 38 already precompiled.

```

```
1 import PlotlyBase, PlotlyKaleido
```

tangent_line (generic function with 1 method)

```
1 function tangent_line(f, x0, x1, c; n = 100)
2     x = range(x0, stop = x1, length = n)
3     df(x) = ForwardDiff.derivative(f, x)
4     plot(x, f, color = Colors.JULIA_LOGO_COLORS.blue, label = "f(x)", ylims =
5         (minimum(f.(x) + df.(x)), maximum(f.(x) + df.(x))), legend=:bottomleft)
6     plot!(x, df, color = Colors.JULIA_LOGO_COLORS.purple, label = "f'(x)")
7     pente = df(c)
8     tangente(x) = pente * (x - c) + f(c)
9     x_right = range(c, stop = c + 1, length = 2)
10    plot!([c, c + 1], tangente, color = Colors.JULIA_LOGO_COLORS.green, label = "")
11    plot!([c, c + 1], [f(c), f(c)], color = Colors.JULIA_LOGO_COLORS.green, label =
12        "")
13    plot!([c + 1, c + 1], [f(c), tangente(c + 1)], color =
14        Colors.JULIA_LOGO_COLORS.red, label = "")
15    scatter!([c], [pente], markerstrokewidth = 0, color =
16        Colors.JULIA_LOGO_COLORS.red, label = "")
17 end
```