

L'usage du code source pour la recherche

Avantages, Inconvénient, Nécessité

Anaïs Vignoles

Aymeric Hermann



Coder vs “pointer-cliquer”

- L'utilisation des logiciels “pointer-cliquer” est facile et intuitive au premier abord, mais se révèle être vite limitée avec des manières de faire très normalisées
- Les logiciels “pointer-cliquer” sont payants et fonctionnent avec des formats de fichiers qui sont peu (ou pas) inter-opérables avec d'autres logiciels (pensez à vos fichiers .xls, .xlsx, .word, .ai, .ps, etc.)

Coder vs “pointer-cliquer”

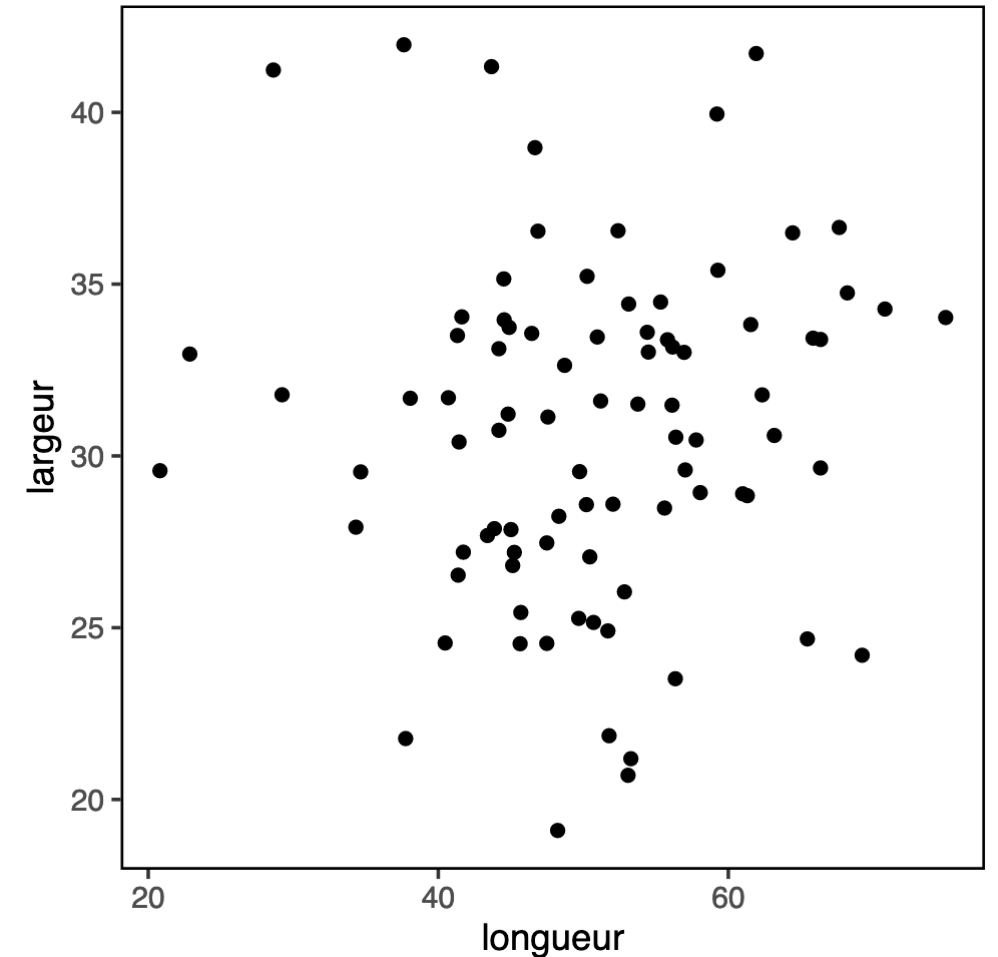
- Tous les logiciels fonctionnent avec un code source, mais la plupart du temps il n'est pas visible pour l'utilisateur. De ce fait, il est difficile de reproduire le chemin parcouru au cours de l'analyse
- Coder veut dire se libérer du pointer-cliquer : expliciter et archiver le détail des interactions avec les données pour le travail dans l'intime, mais aussi pour partager un travail finalisé et mis au propre...

Coder vs “pointer-cliquer”

Les particularités de la programmation :

- Efficacité et rapidité

```
dataframe %>%  
  ggplot(aes(x = longueur, y = largeur)) +  
  geom_point() +  
  my_theme
```

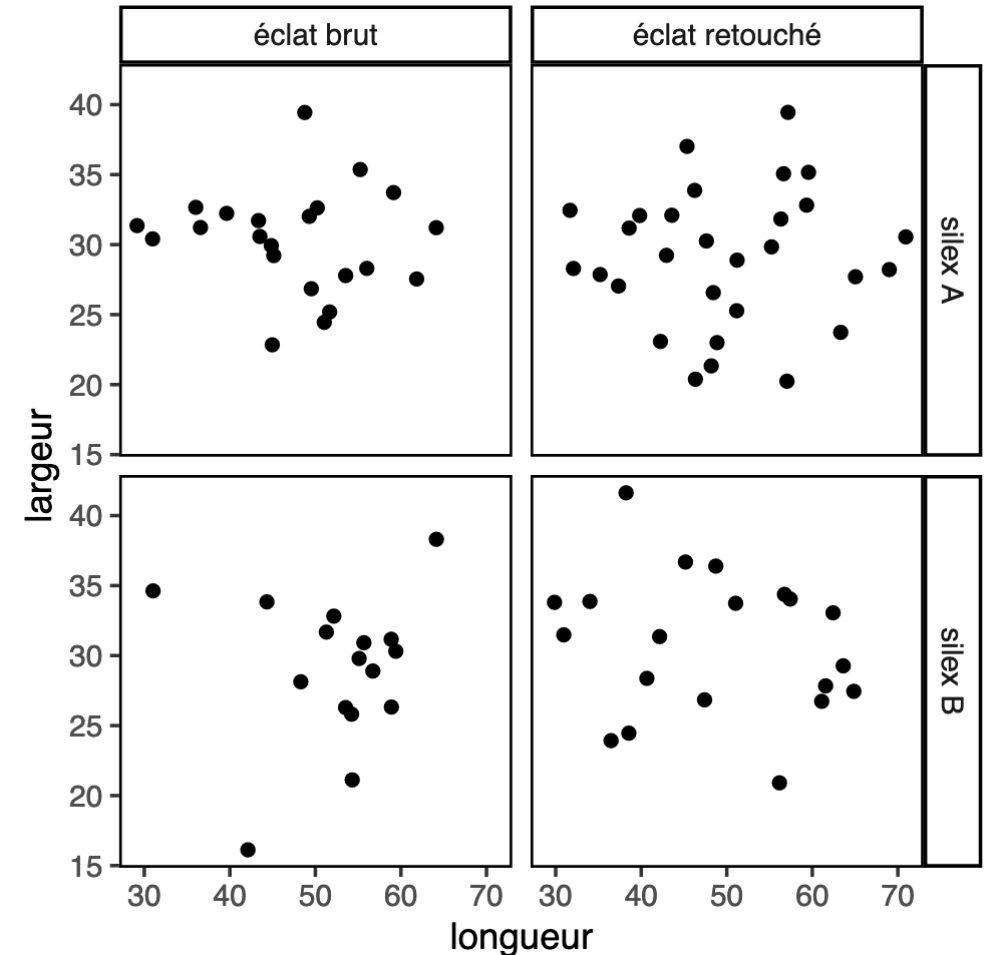


Coder vs “pointer-cliquer”

Les particularités de la programmation :

- Efficacité et rapidité
- Itération

```
dataframe %>%  
  ggplot(aes(x = longueur, y = largeur)) +  
  geom_point() +  
  my_theme +  
  facet_grid(cols = vars(categorie),  
             rows = vars(matiere_premiere))
```



Coder vs “pointer-cliquer”

Les particularités de la programmation :

- Efficacité et rapidité
- Itération
- Possibilité de partager (“copier-coller”)



Coder vs “pointer-cliquer”

Les particularités de la programmation :

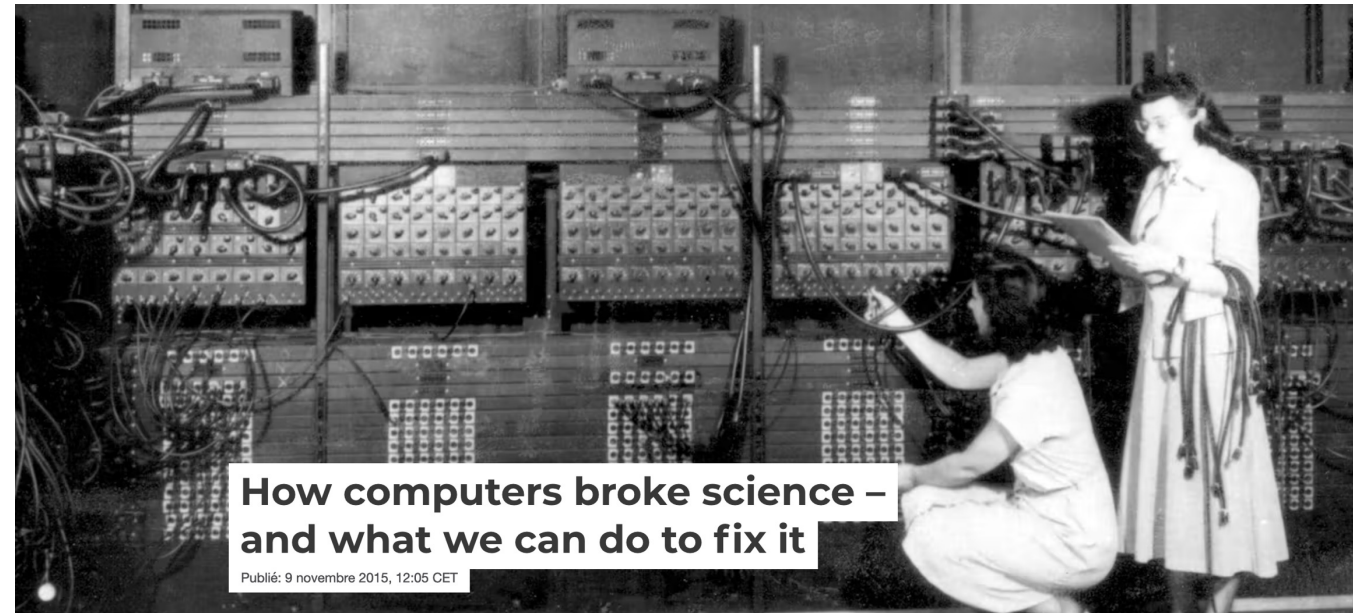
- Efficacité et rapidité
- Itération
- Possibilité de partager
- Maîtriser la machine informatique

rage against the machine

THE CONVERSATION



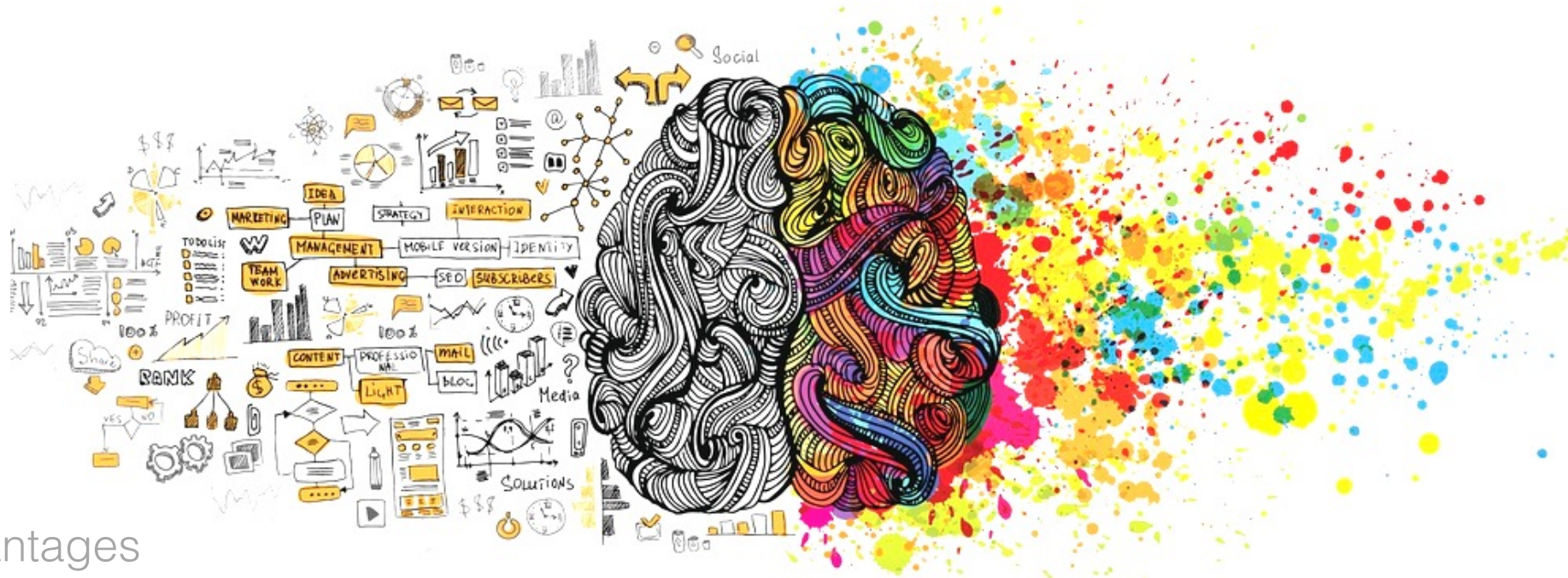
Ben Marwick
Associate Professor of
Archaeology, University of
Washington



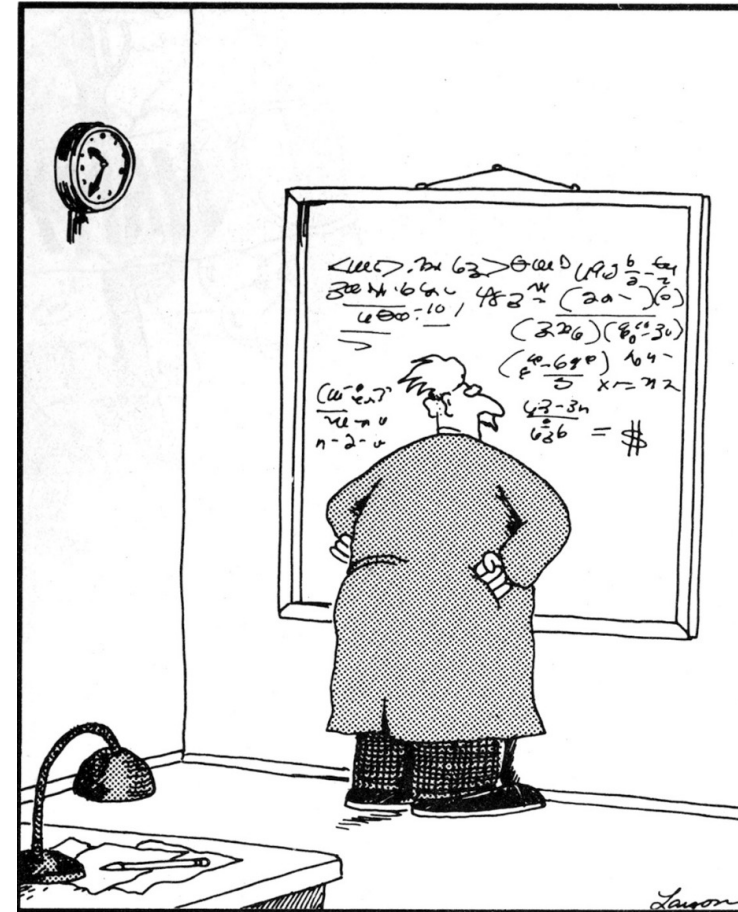
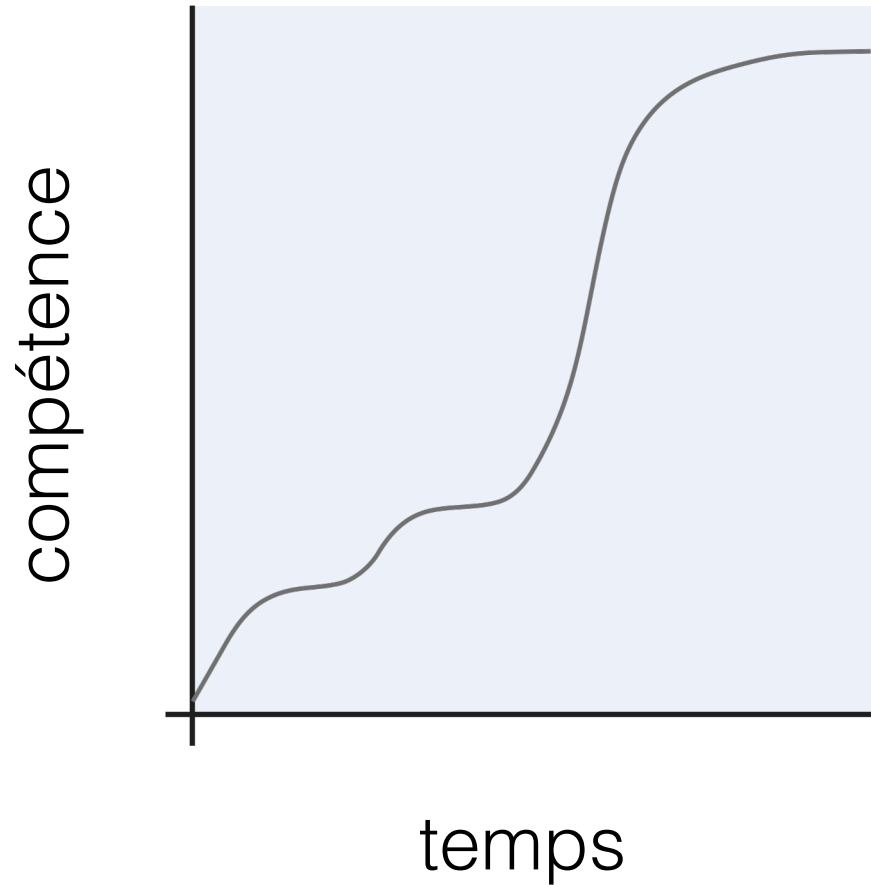
Coder vs “pointer-cliquer”

Les particularités de la programmation :

- Restaure et promeut la créativité dans le traitement des données : multiples solutions possibles, incite l'anticipation et la rationalisation dans les démarches analytiques, etc.



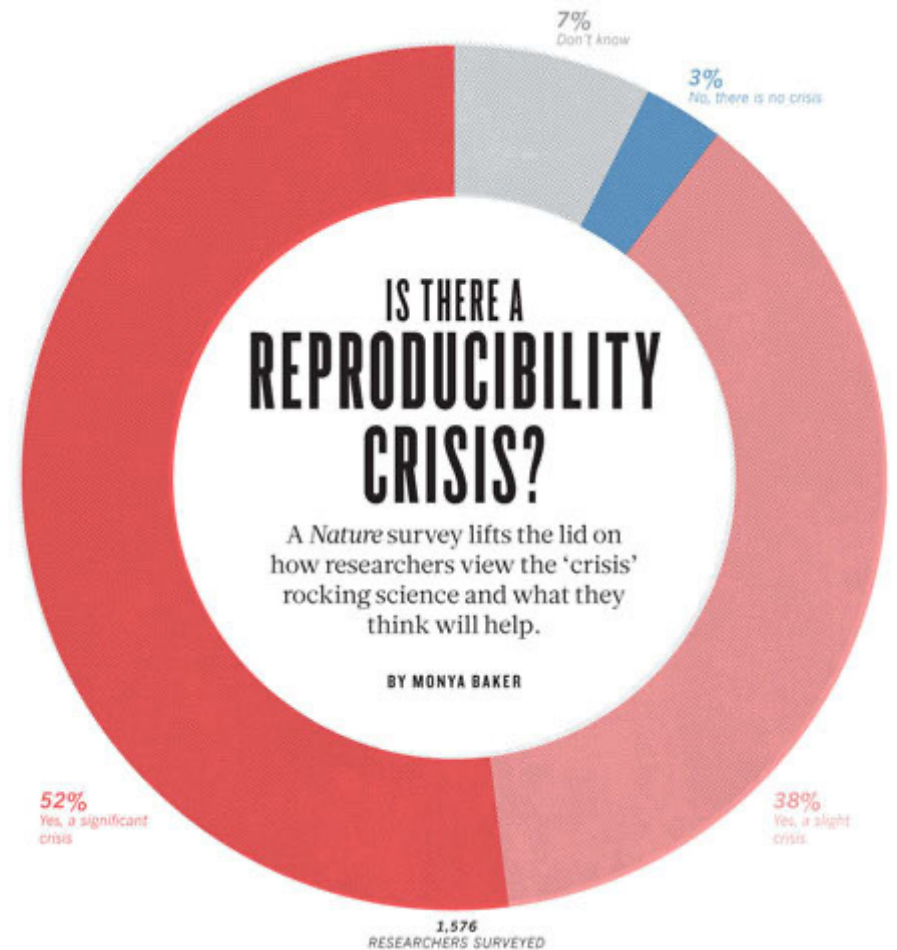
(Encore) Un nouvel outil à maîtriser



Einstein discovers that time is actually money.

Faciliter la reproductibilité

- Reproduire son analyse pour soi, pour s'assurer que le processus de traitement et d'analyse des données est cohérent, pour réutiliser une même approche à l'avenir...
- Reproductibilité pour les autres (relecteurs, collègues, étudiants)
- Endiguer la crise de la reproductibilité lorsque c'est possible



Les principes de la science ouverte : FAIR

www.nature.com/scientificdata

- **Findable**
Faciles à trouver
- **Accessible**
Accessibles
- **Interoperable**
Interopérables
- **Reusable**
Réutilisables

SCIENTIFIC DATA 

Amended: Addendum

OPEN
SUBJECT CATEGORIES
» Research data
» Publication
characteristics

Comment: The FAIR Guiding Principles for scientific data management and stewardship

Mark D. Wilkinson *et al.*[#]

Les principes de la science ouverte

Science ouverte – Codes et logiciels

Ministère de l'Enseignement supérieur et de la
Recherche

Août 2022

Coordination éditoriale : Université de Lille

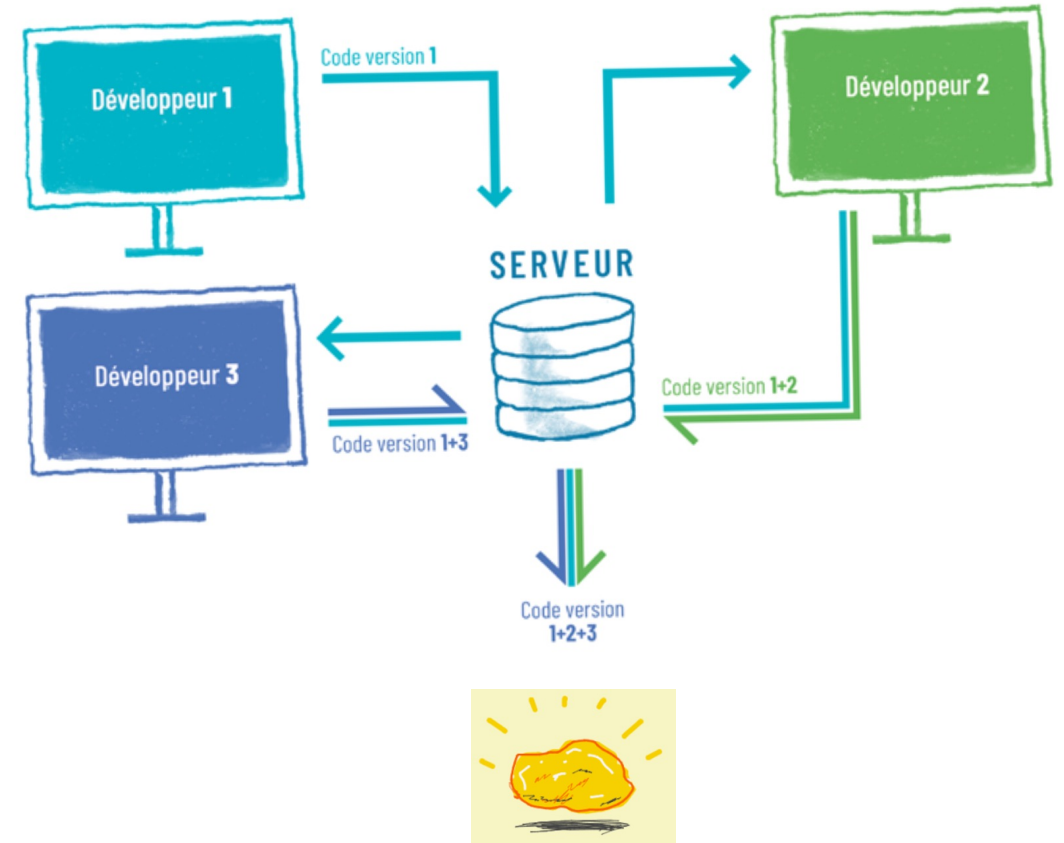
[Accéder à la version numérique du guide](#)



Les principes de la science ouverte

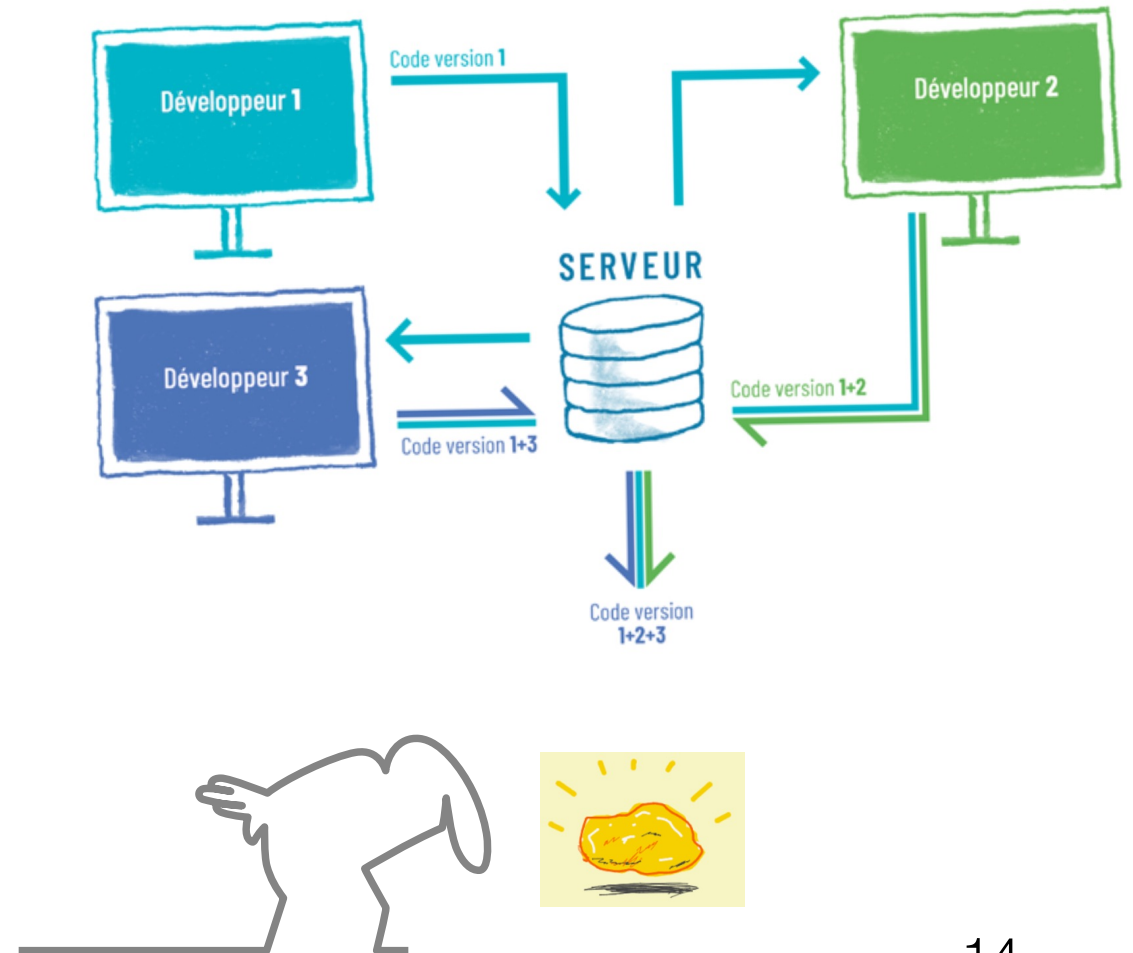
- Le travail direct avec code source rend le travail collaboratif plus facile et efficace

Système de “forge” utilisant la technologie **git** qui est mis à disposition par Huma-Num



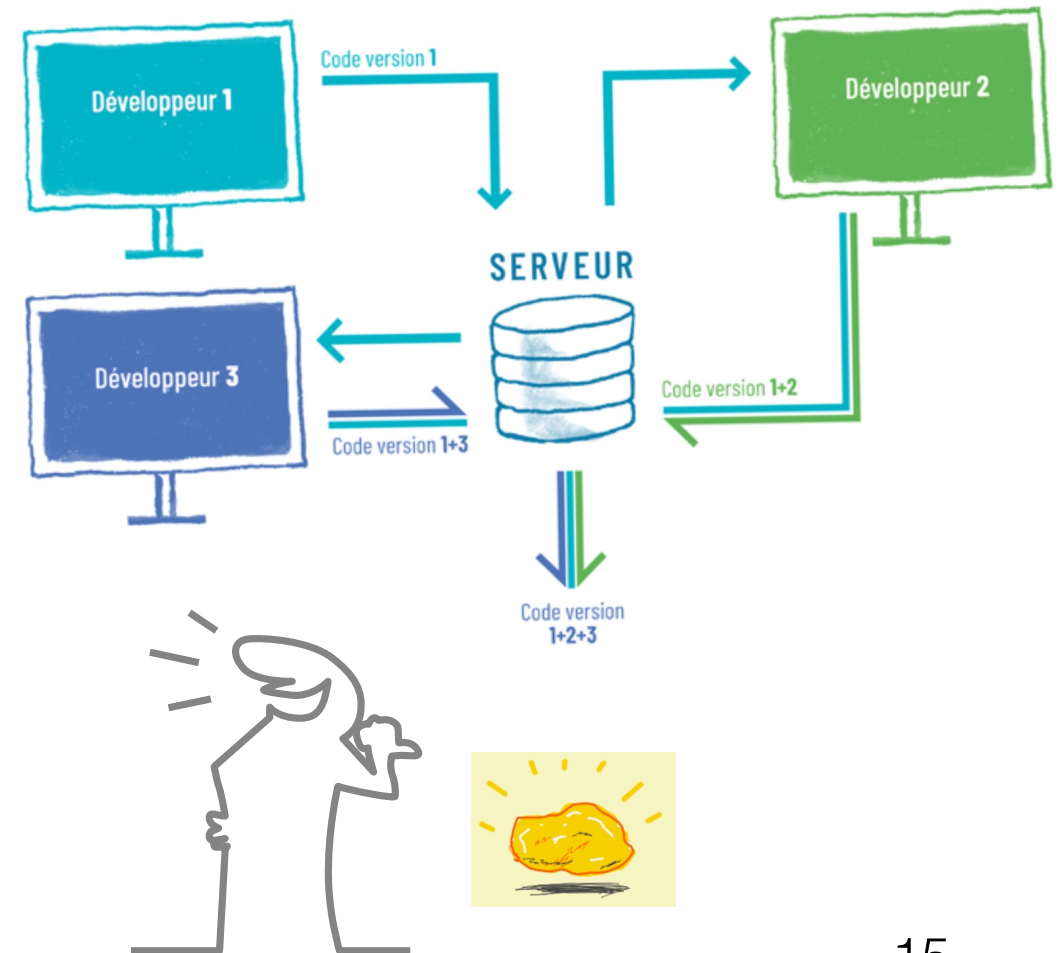
Les principes de la science ouverte

- Le travail direct avec code source rend le travail collaboratif plus facile et efficace
- Le partage du code source permet de valoriser le travail en révélant les “manières de faire”



Les principes de la science ouverte

- Le travail direct avec code source rend le travail collaboratif plus facile et efficace
- Le partage du code source permet de valoriser le travail en révélant les “manières de faire”
- Rendre accessible et promouvoir un certain type d'analyse ou approche méthodologique particulière



Plan national de la science ouverte

- 1^{er} plan (2018-2021) :
 - Création d'un Comité pour la science ouverte
 - Soutient des initiatives majeures concernant publications et données
- 2^{ème} plan (2021-2024) :
 - Vise à généraliser les pratiques de sciences ouverte, à partager et ouvrir les données de la recherche, promouvoir les codes sources produits
 - Fixe comme objectif 100% des publications en accès ouvert en 2030
 - Moyens alloués à ce programme : le budget passe de 5M€ à 15M€ par an
 - Création d'une plateforme nationale des données de la recherche
 - Valorisation et soutien à la diffusion des codes sources sous licence libre pour les travaux financés sur fonds publics

En somme, pourquoi apprendre à utiliser R ?

- D'abord, d'un point de vue très pragmatique, **pour gagner du temps** (à terme)
- **Faciliter le travail collaboratif** (environnement de travail commun à tous, outil non payant à l'inverse d'excel, ArcGis, SPSS, etc.)
- Pour **faciliter l'adoption de bonnes pratiques** pour une Science Ouverte (processus de travail transparents, ouverts et faciles à répliquer)

Pour plus d'informations sur R



Introduction à R : Base du langage, packages, Rstudio et documentation, par H. Pecout



Installer R et RStudio (presque !) sans peine, par A. Perdoncin



Une introduction à R et au tidyverse, par J. Barnier



Découvrir R et Rstudio, par T. Zorn et al.

Workshop « R pour les archéologues ! »



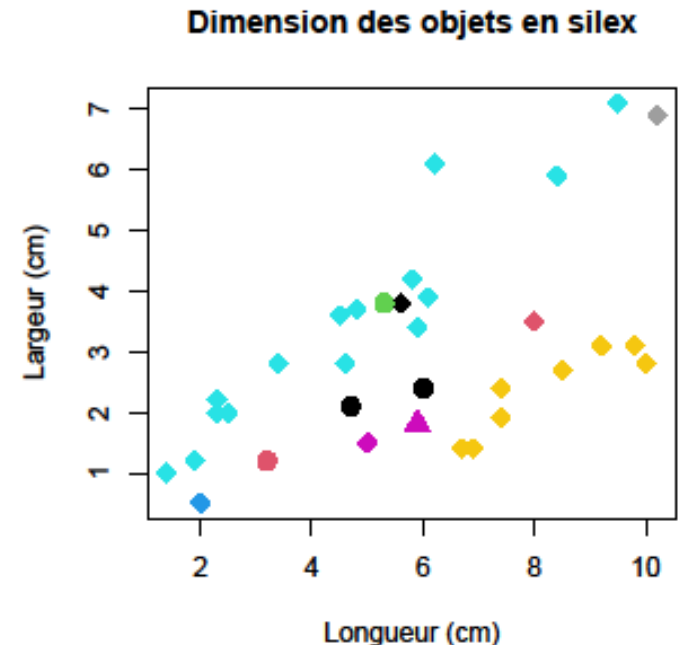
Séance 1 : Se familiariser avec le langage de programmation R

Objectif :

Se familiariser avec le langage R, sa syntaxe et l'interface Rstudio

```
Seance1.R x
Source on Save
Run
Source

20
21 #####
22 # Ouvrir un jeu de données ####
23 #####
24
25 #####
26 # Nous allons commencer par ouvrir le jeu de données "Data_exemple" en utilisant
27 # le bouton "Import dataset" dans l'onglet "Environnement" en haut à droite. Ce
28 # bouton permet d'écrire automatiquement le code pour importer un tableau excel.
29 jdd <- Data_exemple
30
31 ## Nous verrons à la fin de la séance comment importer des données avec une ligne
32 ## de code directement (plus reproductible !).
33
34 #####
35 # Vous venez d'ouvrir le tableau dans R ! Il a été stocké dans un objet que l'on
36 # a appelé "jdd". Si vous l'appellez dans la console, il s'affichera, vous
37 # permettant de l'explorer.
38 jdd
39
```



Du code à la figure...

Séance 2 : Analyser et visualiser un jeu de données semi-quantitatif

Objectifs :

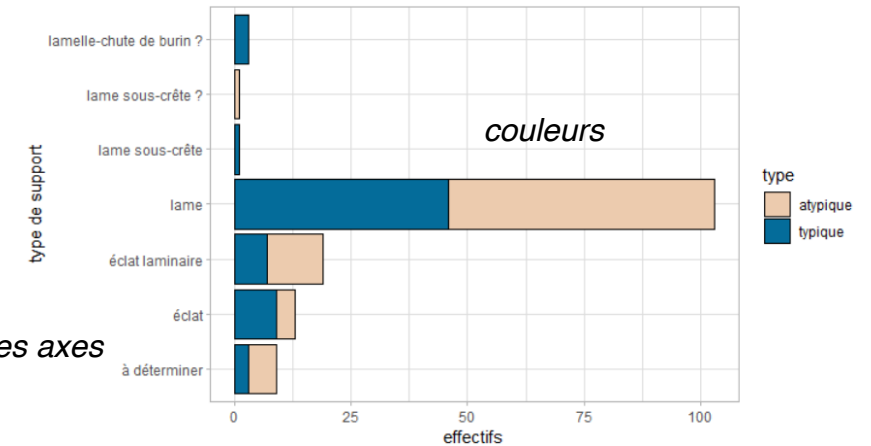
- Explorer et mettre en forme ses données (package tidyverse)
- Créer de belles figures (package ggplot)

...Faire varier l'esthétique des graphiques

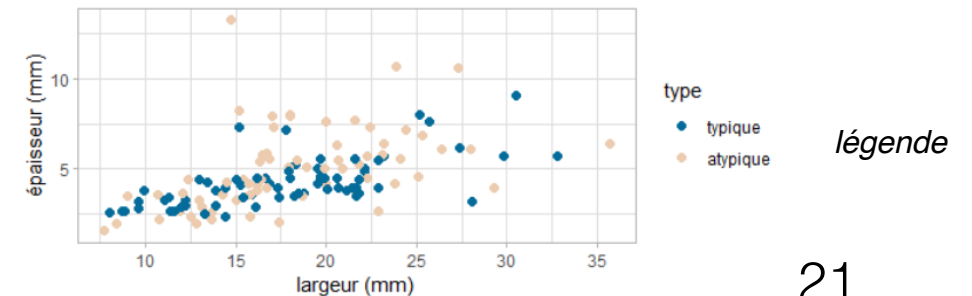
| | année | couche | carré | numéro | type | type_support | long | larg | e | morphologie |
|----|-------|--------|------------|--------|----------|-------------------|---------|---------|-----|-------------|
| 1 | 1959 | 11 | 80 | 1557 | typique | lame | 30.5 | 18.3 | 5.3 | simple |
| 2 | 1960 | 11 | 81 | 538 | typique | lame | non mes | 18.2 | 3.5 | simple |
| 3 | NA | 11c | 82/83 | NA | typique | à déterminer | non mes | non mes | 1.5 | simple |
| 4 | 1959 | 11 | 80 | 1558 | atypique | lame | non mes | non mes | 2.6 | simple |
| 5 | 1959 | 11 | 80 | 1600 | typique | lame | non mes | 23.2 | 5.7 | simple |
| 6 | 1959 | 11 | 83 | 1157 | atypique | lame | 26.6 | 21.7 | 3.6 | simple |
| 7 | 1960 | 11 | chantier J | NA | atypique | lamelle | 19.2 | 7.7 | 1.6 | symétrique |
| 8 | 1959 | 11a | 82 | NA | atypique | éclat laminaire | 32.8 | 29.3 | 4.0 | opposé |
| 9 | 1959 | 10 | 77 | 294 | typique | éclat laminaire | 27 | non mes | 6.1 | simple |
| 10 | 1959 | 11c | 83? | 1536 | atypique | lame ? | 38.8 | 12.8 | 2.0 | alterne |
| 11 | 1959 | 11c | 83 | 1402 | atypique | éclat laminaire ? | 27.0 | 17.1 | 7.3 | simple |

Showing 1 to 11 of 149 entries, 20 total columns

labels des axes



type de graphique



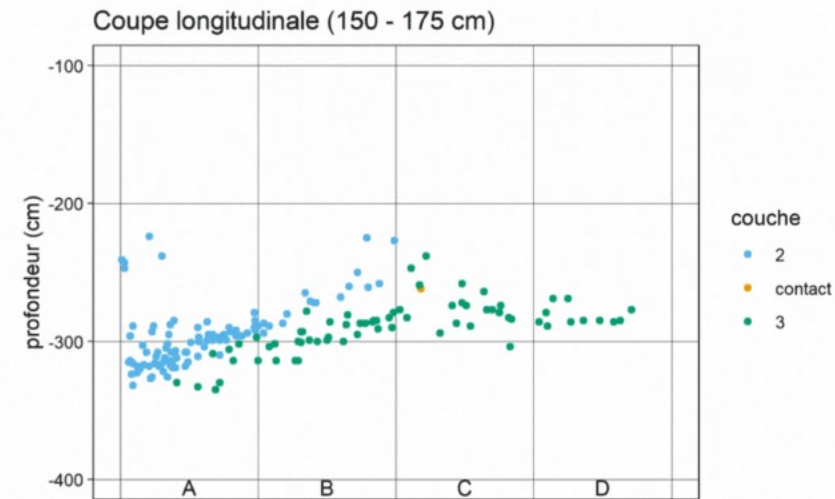
Séance 3 : Projections spatiales automatisées

Objectif : se familiariser avec le concept de « boucles » en programmation

Appliquer la même opérations sur des jeux de données différents...

```
# loop pour chaque projection
for(i in 1:i.nb) {

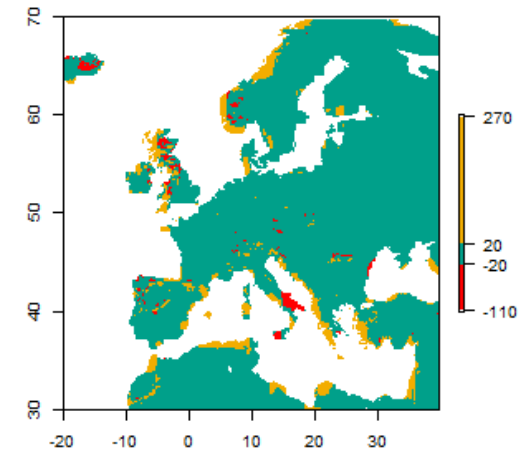
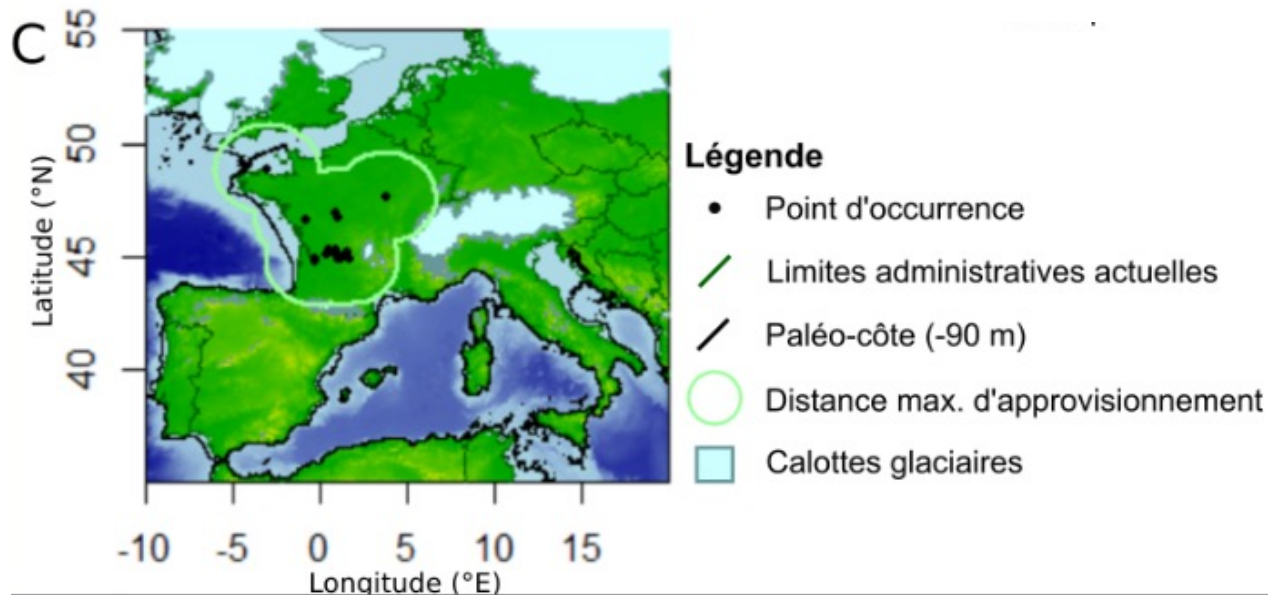
  # projection avec ggplot2
  proj <- ggplot2::ggplot() +
    theme_linedraw() +
    geom_point(data = tranche, mapping = aes(x = x, y = y, color = couche)) +
    coord_fixed(ratio = 1, xlim = c(0, 400), ylim = c(-400, -100)) +
    theme(axis.text.x = element_blank(),
          panel.grid.major = element_line(),
          panel.grid.minor = element_blank()) +
    scale_x_continuous(breaks = seq(0, max, by = 100)) +
    labs(title = titre, y = "profondeur (cm)", x = element_blank()) +
    scale_color_manual(values = couches_colors) +
    annotate(geom = "text", x = seq(50, 350, by = 100), y = -406, label = carrés)
```



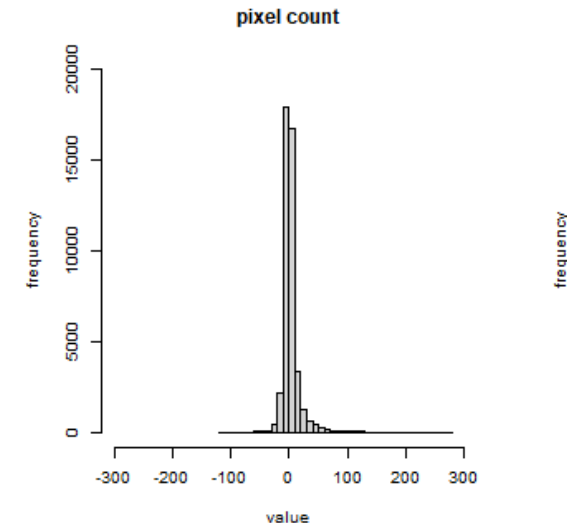
Séance 4 : Les SIG sous R

Objectifs :

- manipuler des données raster et vector
- créer une carte
- analyser des données spatialisées (intro)



Soustraction de rasters...



... décompte de pixels

Organisation du workshop

- Pour participer contactez Anaïs Vignoles (anaïs.l.vignoles@gmail.com) et Aymeric Hermann (aymeric.hermann@cncrs.fr)
- 4 séances de 2h chacune (1h consacrée à la découverte d'un ensemble de codes, et 1h de mise en situation)
- Les 4 séances auront lieu à la MSH Mondes pendant les deux dernières semaines d'Avril (semaines 16 et 17). L'heure et le lieu restent à définir.
- En présentiel uniquement. La documentation et les ressources utilisées seront partagées après chaque séance pour celles et ceux qui ne peuvent y assister