Zen programming

Situations

You have a deadline tomorrow and prepare to leave at 18h30.
At 18h29, you add a label to a plot and get :

```
0x1016e8f20. This is a serious error. This application, or a library it uses, is using an
invalid context and is thereby contributing to an overall degradation of system stability
and reliability
```

Situations

You have a deadline tomorrow and prepare to leave at 18h30.
At 18h29, you add a label to a plot and get :

```
0x1016e8f20. This is a serious error. This application, or a library it uses, is using an
invalid context and is thereby contributing to an overall degradation of system stability
and reliability
```

You are going to a workshop and planning to show your new, shiny R package you developed with *Bighard Revolution pro-R™* and the R package *coolstuffr* from last week.

Nobody manages to install it.

Situations

You don't care about all those fancy new software, you're an ecologist after all. You installed R some years ago and do stuff you consider basic with it.

- The rightmost break for the "months", "quarters" and "years" cases of hist.POSIXlt() has been increased by a day. (Inter alia, fixes PR#15717.)
- The handling of DF[i,] <- a where i is of length 0 is improved. (Inter alia, fixes PR#15718.)

April 2014

- hclust() gains a new method "ward.D2" which implements Ward's method correctly. The previous "ward" method is "ward.D" now, with the old name still working. Thanks to research and proposals by Pierre Legendre.
- The sunspot.month dataset has been amended and updated from the official source, whereas the sunspots and sunspot.year datasets will remain immutable. The documentation and source links have been updated correspondingly.

Situations

You manage a small group of people doing weather forecasting.
Although people work together in discussions and meetings, the code modelling rain has five different implementations.

You have a look and cannot understand any of them.

Situations

You manage a small group of people doing weather forecasting.
Although people work together in discussions and meetings, the code modelling rain has five different implementations.

You have a look and cannot understand any of them.

You manage yourself.

You have a look at your code and cannot understand it.

Problems ?

It's not only about you !

Problems ?

Change one thing: everything breaks

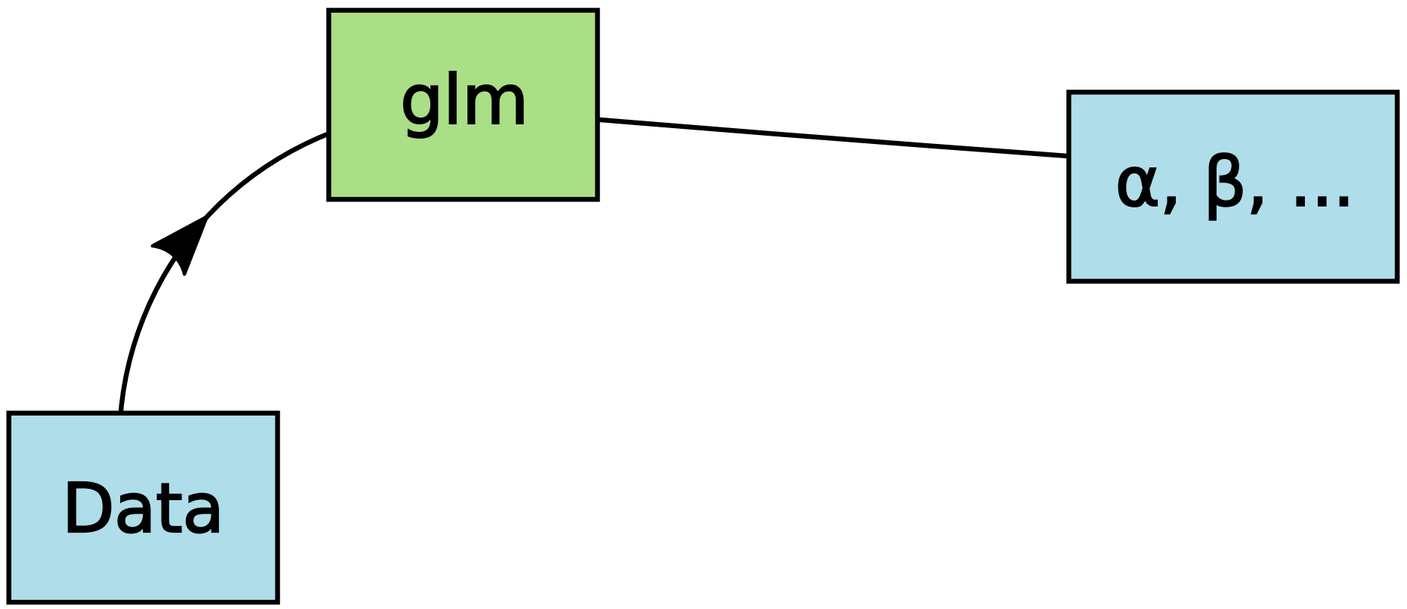
Something stops working at the last minute:
no way to reverse changes

Cryptic, untractable errors

Things never work on others' computers

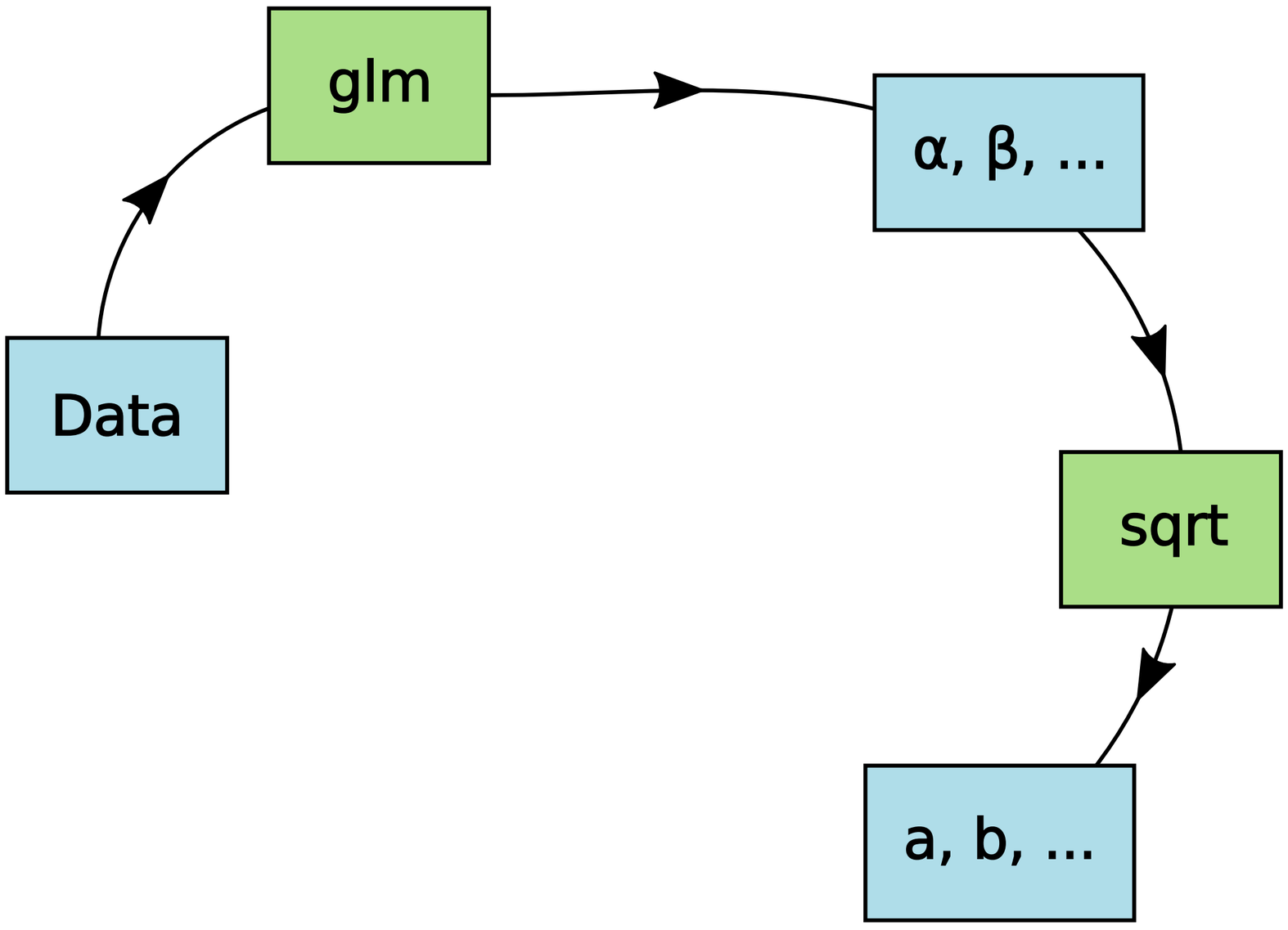
Unsure about what your program does

Why do we program ?

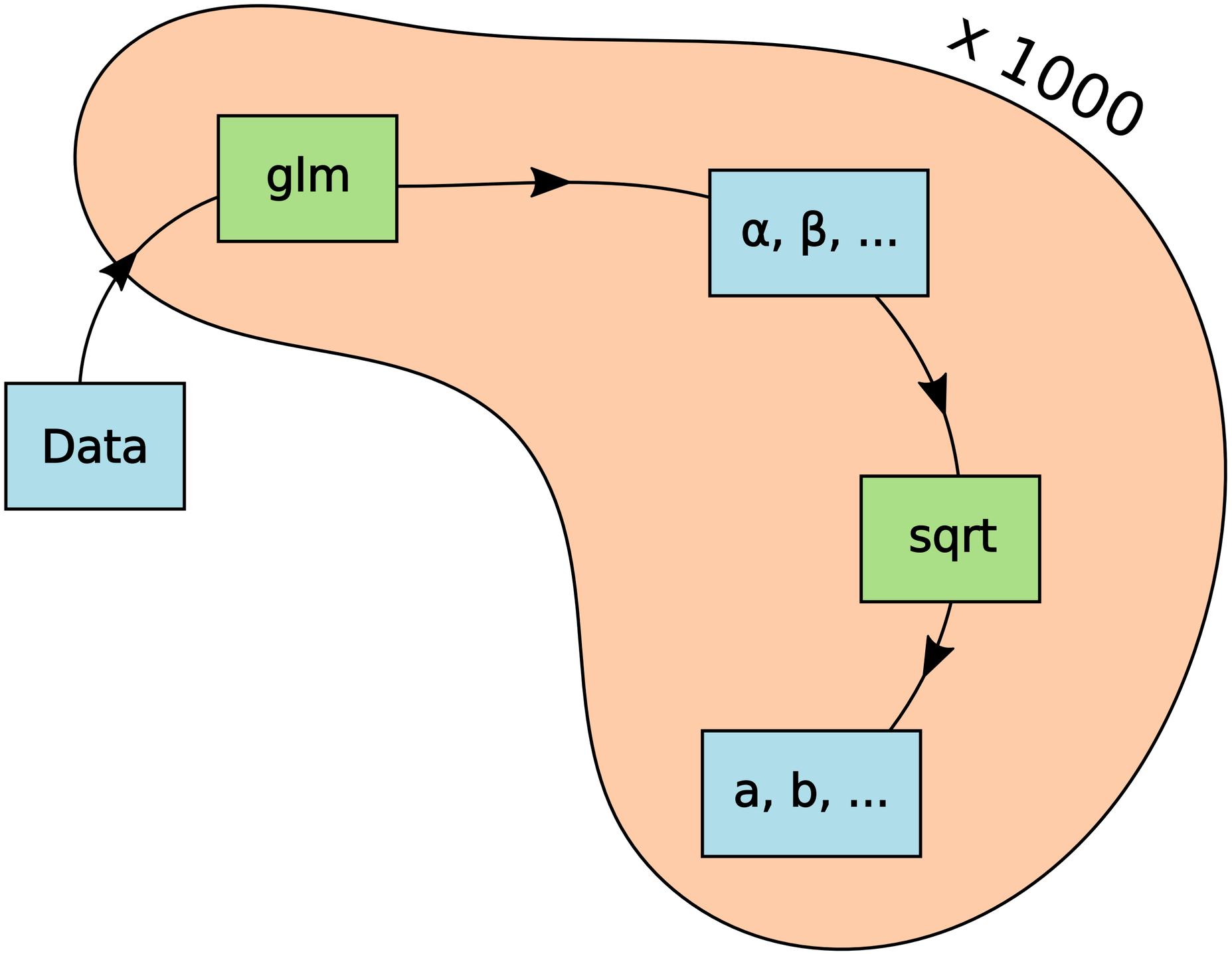


Why do we program ?

Why do we program ?

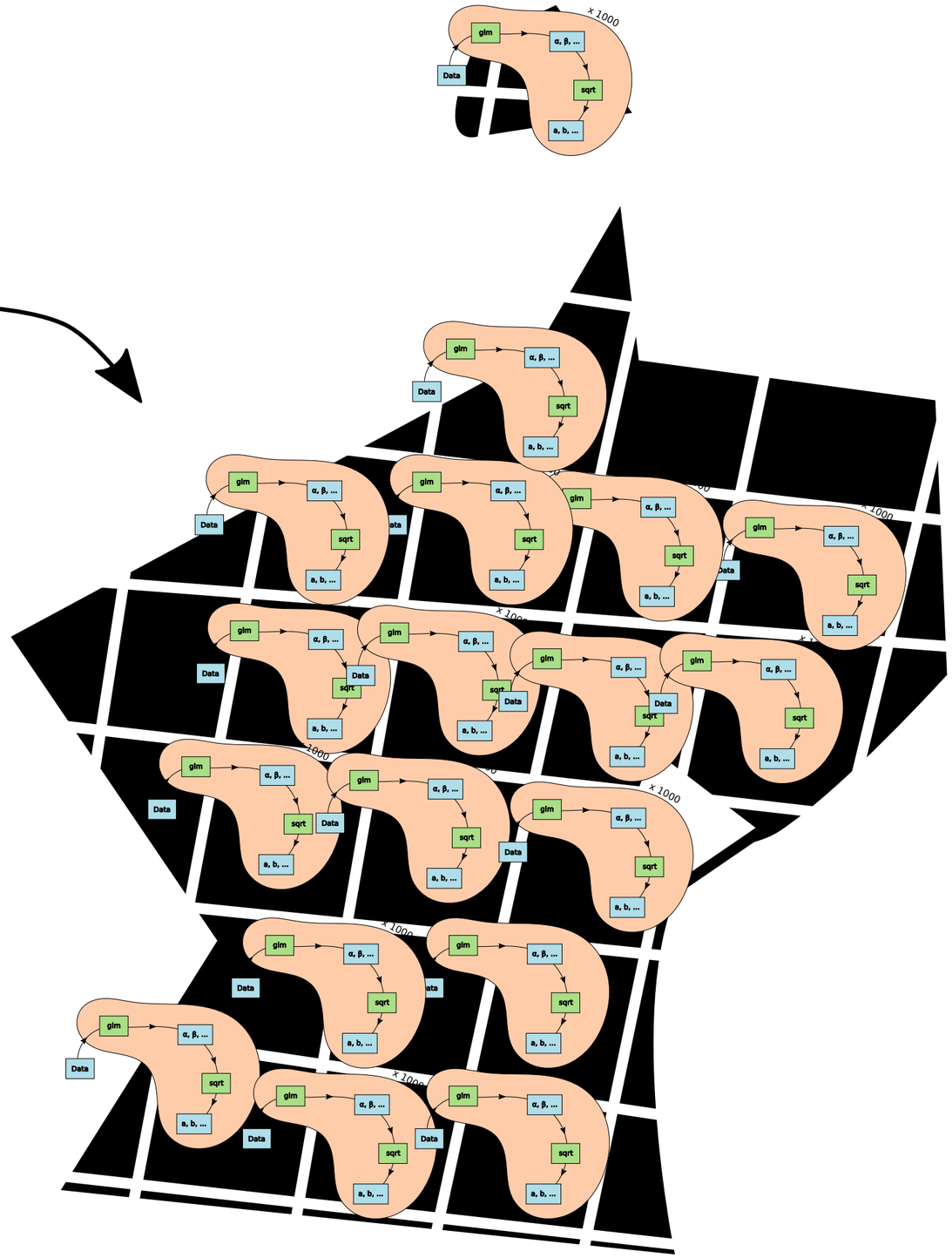


Why do we program ?



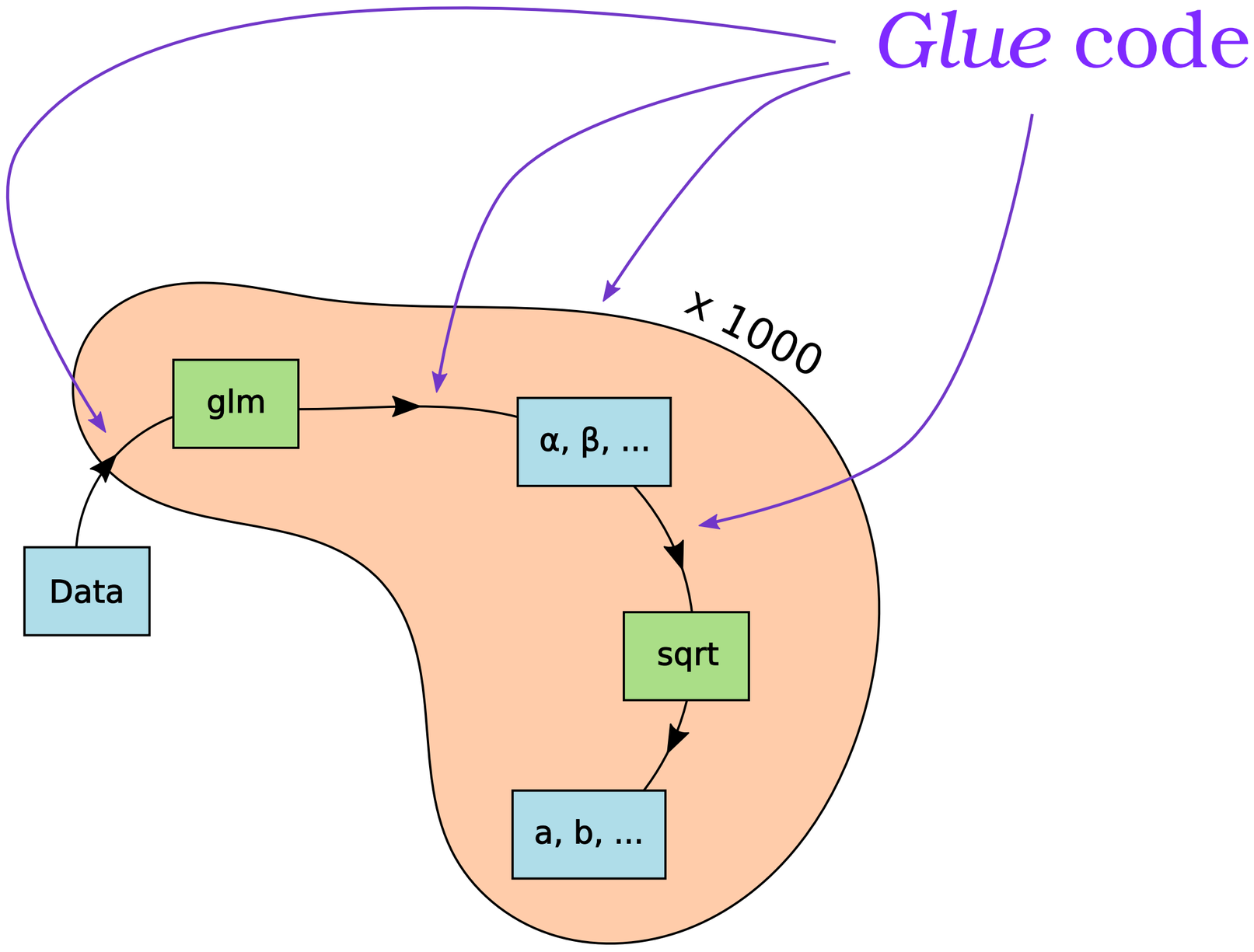
Why do we program ?

your favorite country



Why do we program ?

for *combining* tools



Why R ?

< this page is intentionnaly left blank >

Five ways to zen programming

Use the right programming paradigm

Recognize programming patterns

Write readable code

Choose the right data structures

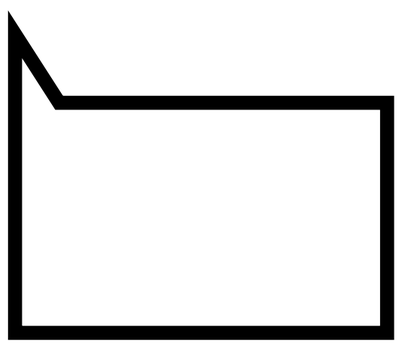
Organize your workspace

Organize your files !

One project - one folder - one good name

Organize your files !

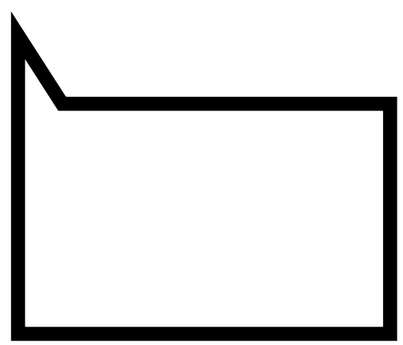
One project - one folder - one good name



Projet: "œil de tigre"

Organize your files !

One project - one folder - one good name



meet_nerdtalk

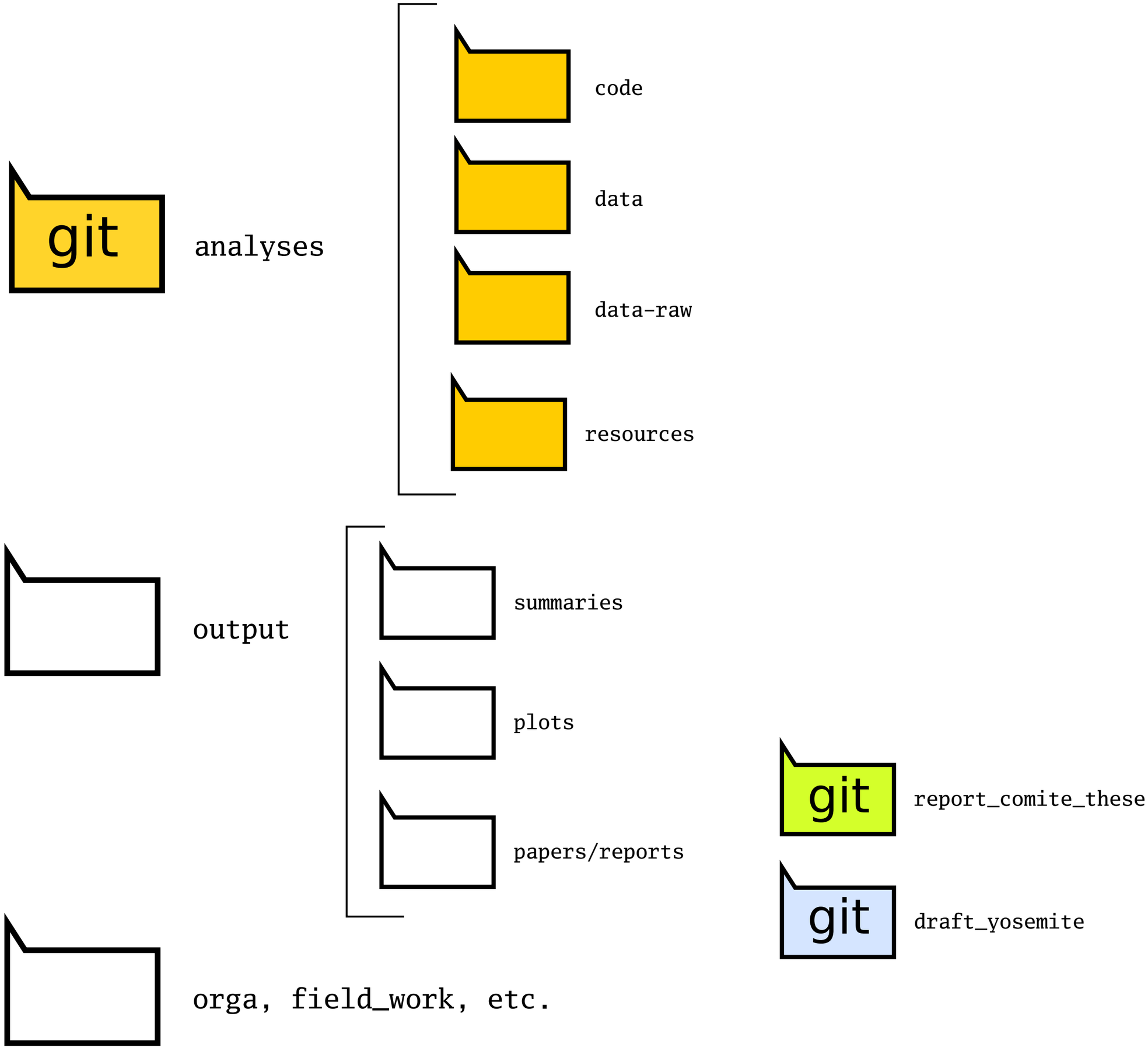
```
C:\Documents and settings\alex\Mes Documents\work\2016\meet_nerdtalk      68 characters

/home/alex/work/2016/meet_nerdtalk                                       34 characters
```

Organize your files !

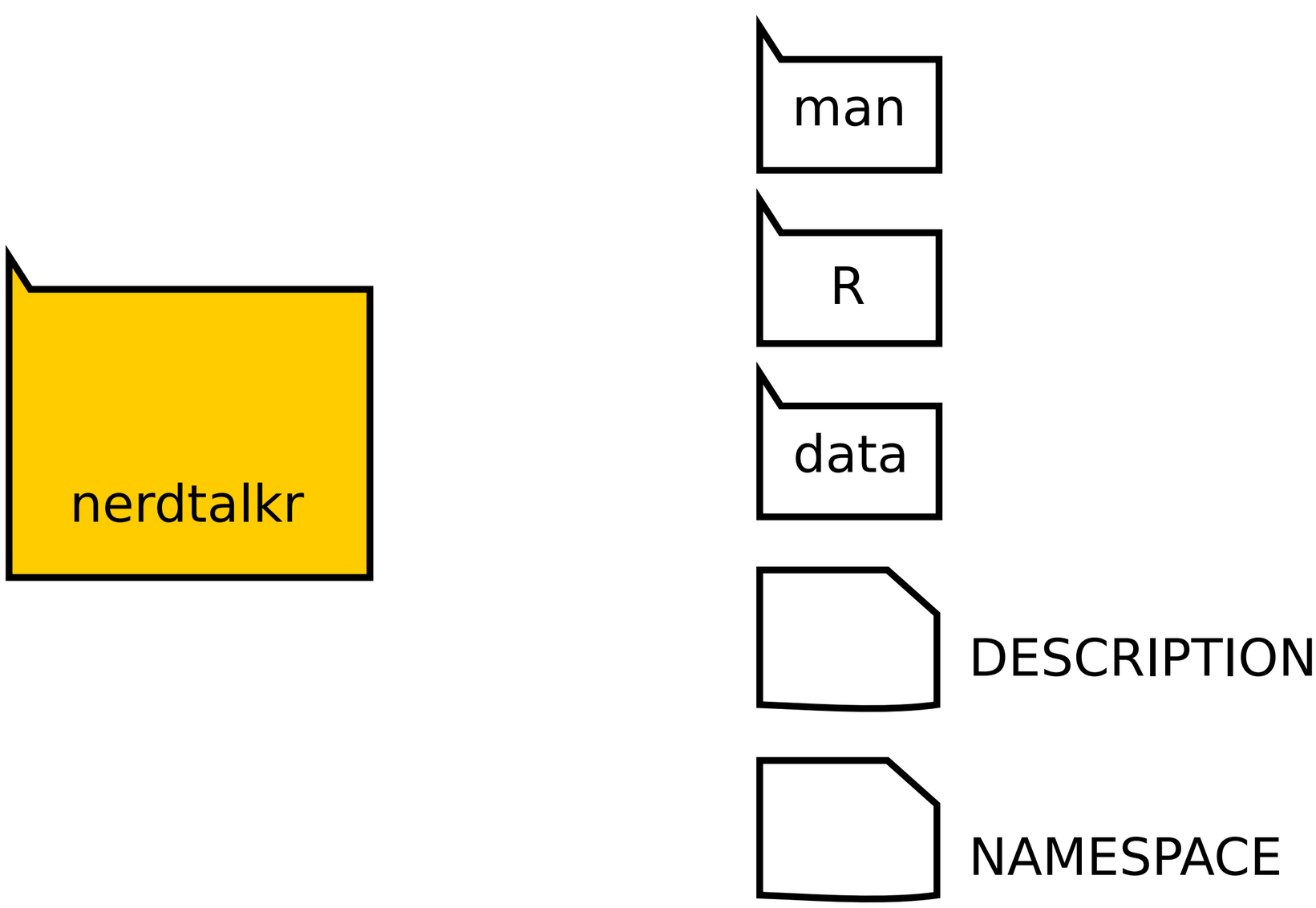
USASCII code chart

b_7 b_6 b_5 → / Bits ↓					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Row ↓ / Column →	0	1	2	3	4	5	6	7
0	0	0	0	0			SP	0	@	P	`	p
0	0	0	1	1			!	1	A	Q	a	q
0	0	1	0	2			"	2	B	R	b	r
0	0	1	1	3			#	3	C	S	c	s
0	1	0	0	4			$	4	D	T	d	t
0	1	0	1	5			%	5	E	U	e	u
0	1	1	0	6			&	6	F	V	f	v
0	1	1	1	7			'	7	G	W	g	w
1	0	0	0	8			(	8	H	X	h	x
1	0	0	1	9			)	9	I	Y	i	y
1	0	1	0	10			*	:	J	Z	j	z
1	0	1	1	11			+	;	K	[	k	{
1	1	0	0	12			,	<	L	\	l	|
1	1	0	1	13			-	=	M	]	m	}
1	1	1	0	14			.	>	N	^	n	~
1	1	1	1	15			/	?	O	_	o	DEL

Organize your files !


git
analyses
code
data
data-raw
resources
output
summaries
plots
papers/reports
orga, field_work, etc.
git
report_comite_these
git
draft_yosemite

Organize your files !

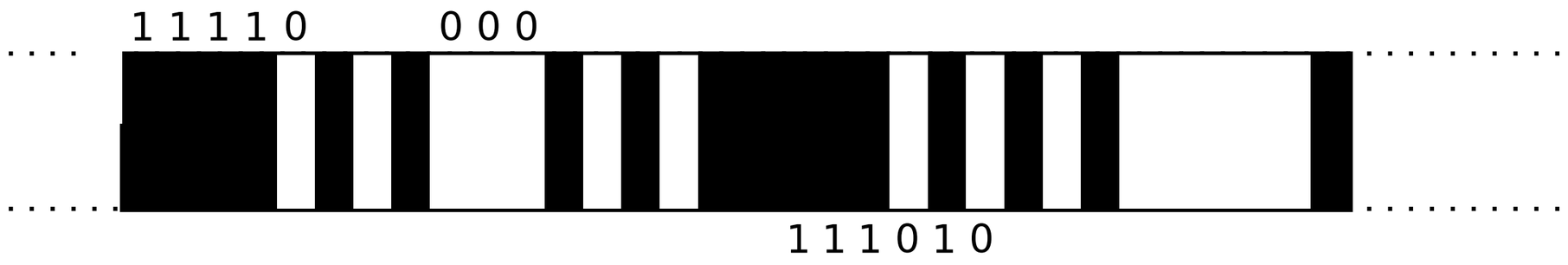


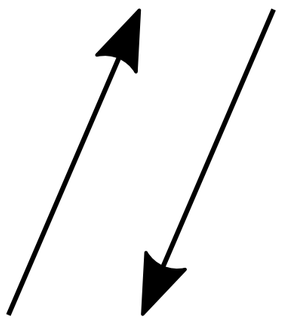
Benefits: Portability checks, unit tests, compiled code,
easy installation on other computers

Data types

A what ?

« A data type in a programming language is a set of data with values having predefined characteristics. »
techtarget.com

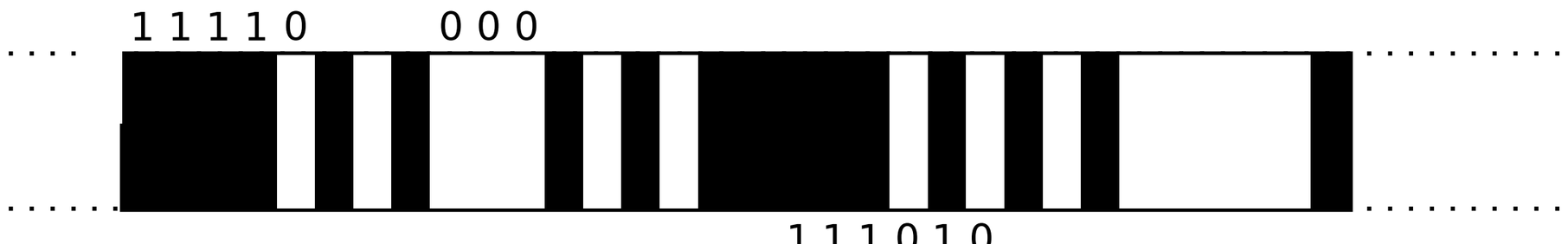


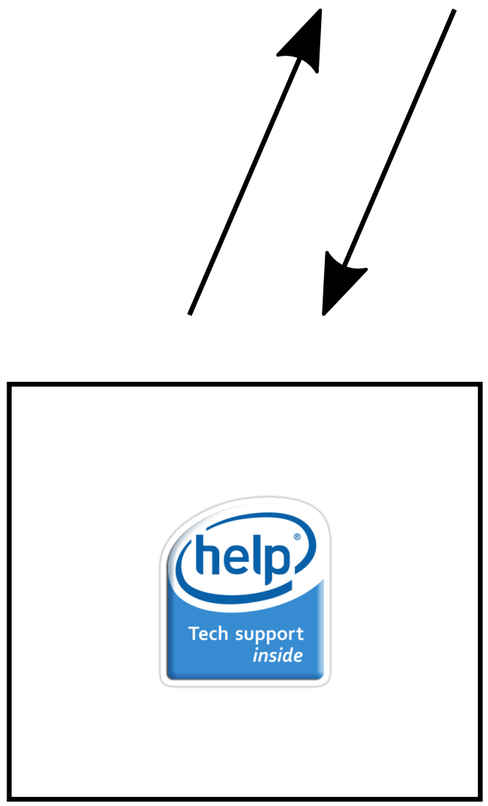


Data types

A what ?

« A data type in a programming language is a set of data with values having predefined characteristics. »
techtarget.com





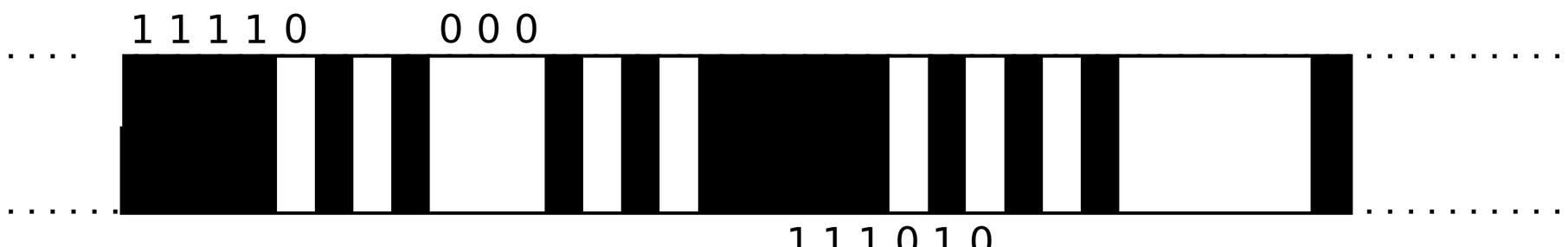

```
111010
```

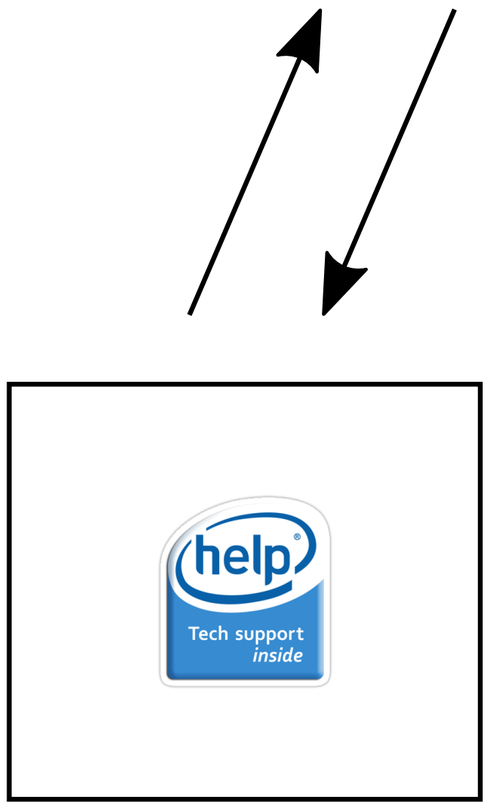
- 1 six-bits integer
- 1 3-bits complex integer

Data types

A what ?

« A data type in a programming language is a set of data with values having predefined characteristics. »
techtarget.com





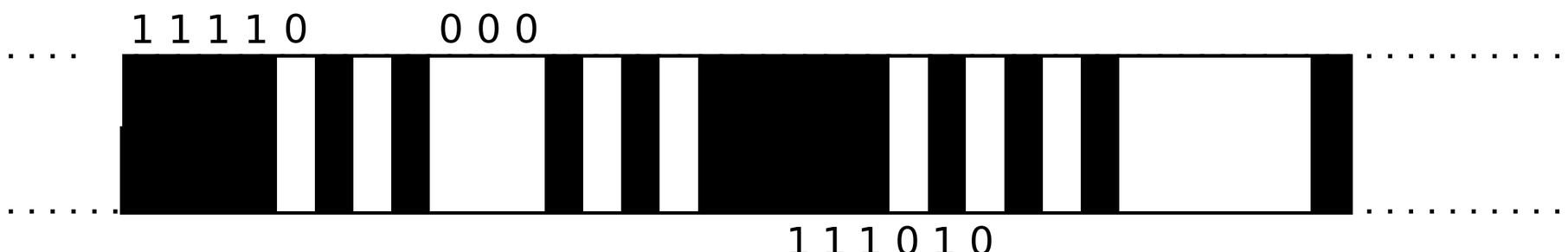

```
111010
```

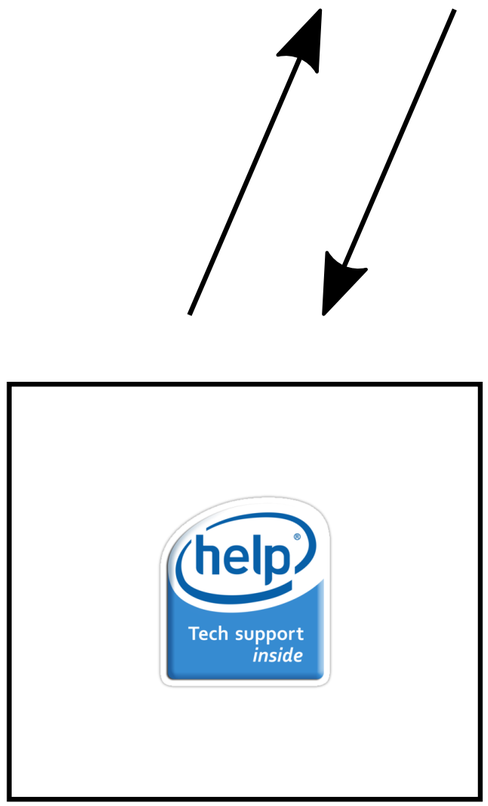
- 1 six-bits integer
- 1 3-bits complex integer
- 1 character (in ASCII)

Data types

A what ?

« A data type in a programming language is a set of data with values having predefined characteristics. »
techtarget.com





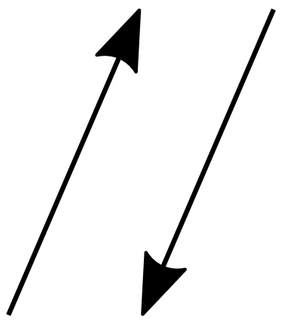

```
111010
```

- 1 six-bits integer
- 1 3-bits complex integer
- 1 character (in ASCII)
- 6 booleans

Data types

A what ?

« A data type in a programming language is a set of data with values having predefined characteristics. »
techtarget.com




```
111010
```

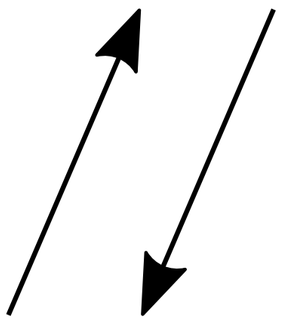
- 1 six-bits integer
- 1 3-bits complex integer
- 1 character (in ASCII)
- 6 booleans

double precision number ?

Data structures

A what ?

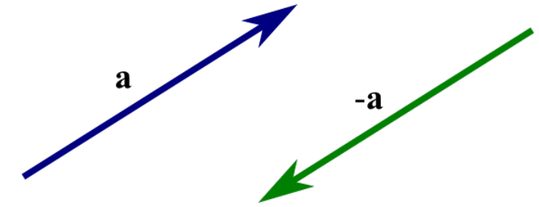
« A data structure is a particular way of organizing data in a computer so that it can be used efficiently »
Wikipedia.

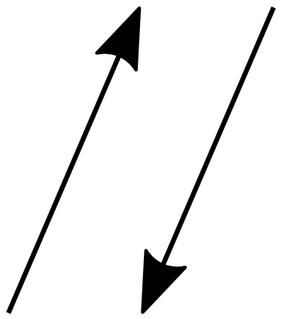


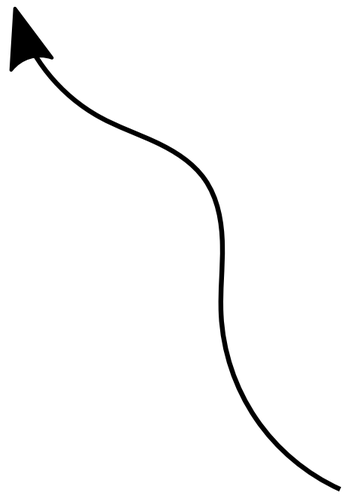
Data structures

A what ?

« A data structure is a particular way of organizing data in a computer so that it can be used efficiently »
Wikipedia.







(1,2)

$$V=\begin{bmatrix} 12 & 7 & 21 & 31 & 11 \\ 45 & -2 & 14 & 27 & 19 \\ -3 & 15 & 36 & 71 & 26 \\ 4 & -13 & 55 & 34 & 15 \end{bmatrix}$$

"the brown fox
jumps over the lazy dog"

Data structures

	homogeneous	heterogeneous
1d	(atomic) vector	list
2d	matrix	data.frame
nd	array	

The coordinates on a surface

The telephone numbers of the lab members

The vegetation surveys in la Crau, France

The vegetation surveys in la Crau, France with a species trait database

	homogeneous	heterogeneous
1d	(atomic) vector	list
2d	matrix	data.frame
nd	array	

The time-series of an ODE model of 100 species, for 1000 simulations

	homogeneous	heterogeneous
1d	(atomic) vector	list
2d	matrix	data.frame
nd	array	

The phylogenetic tree of alpine plants

	homogeneous	heterogeneous
1d	(atomic) vector	list
2d	matrix	data.frame
nd	array	

Data structures

	homogeneous	heterogeneous
1d	(atomic) vector	list
2d	matrix	data.frame
nd	array	

Data structures

	homogeneous	heterogeneous
1d	(atomic) vector [ ]	list [[ ]] $ []
2d	matrix [,]	data.frame [,] $ [[ ]]
nd	array [, ,]	

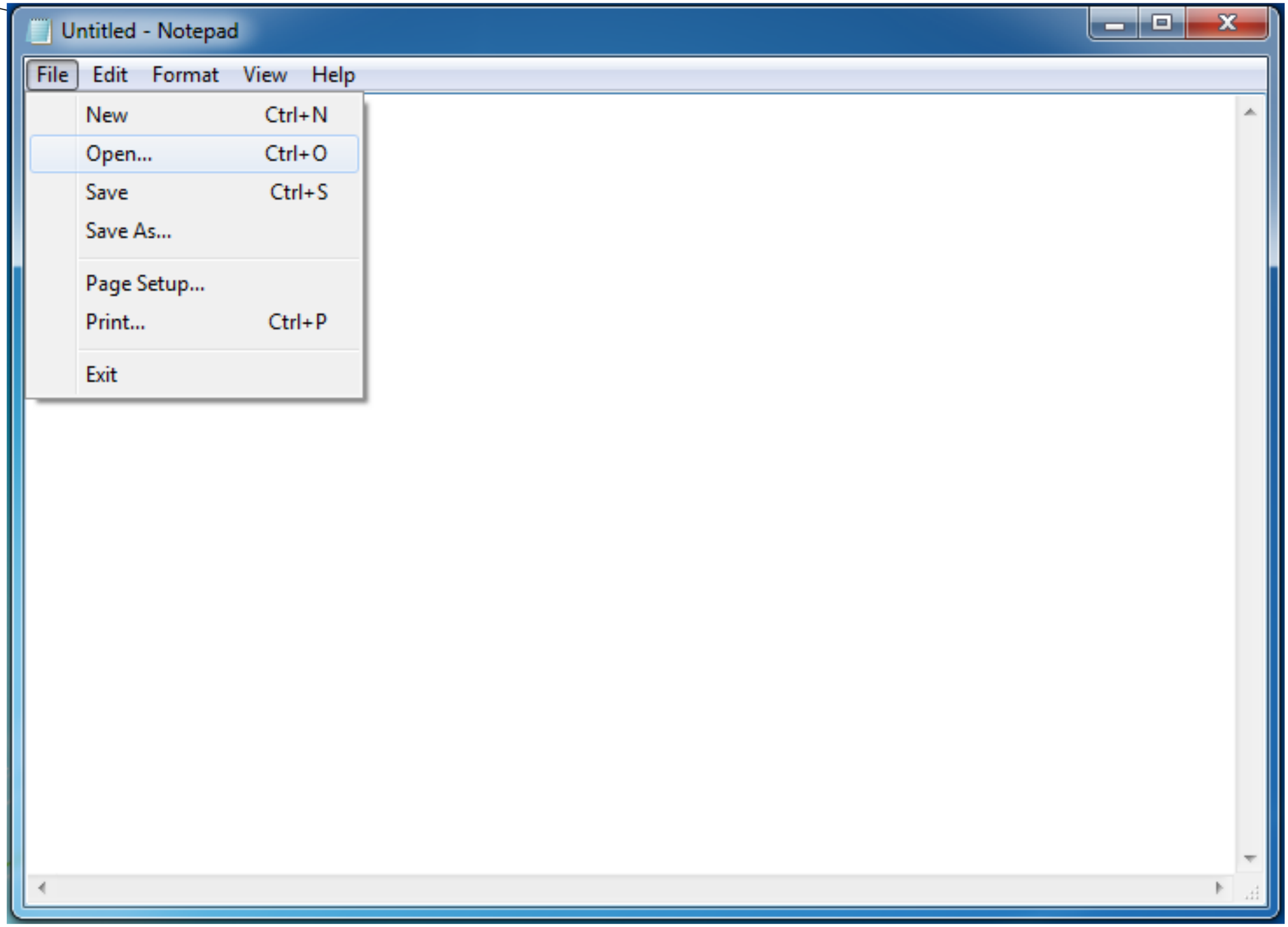
not recommended

Write readable code

A good keyboard for programming

A good text editor

Write readable code



Write readable code

F:\Project Invincible\Project Invincible\core\OnRender.cpp - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

OnInit.cpp | OnRender.cpp | sdl_user_interface.cpp

```
1  #include "sdl_user_interface.h"
2
3  void markSquare(int x,
4      int y,
5      GLfloat red,
6      GLfloat green,
7      GLfloat blue,
8      GLfloat alpha,
9      GLfloat zIndex)
10 {
11     glBegin(GL_QUADS);
12         glColor4f(red, green, blue, alpha);
13         glNormal3i(0, 1, 0);
14         x = 2*x;
15         int z = -2*y;
16         glVertex3f((GLfloat)x-1, zIndex, (GLfloat)z+1);
17         glVertex3f((GLfloat)x+1, zIndex, (GLfloat)z+1);
18         glVertex3f((GLfloat)x+1, zIndex, (GLfloat)z-1);
19         glVertex3f((GLfloat)x-1, zIndex, (GLfloat)z-1);
```

C++ source file length : 5272 lines : 237 Ln : 1 Col : 1 Sel : 0 Dos\Windows ANSI INS

Write readable code

FOLDERS

- marasys-cscart-beta
  - .idea
  - addons
  - controllers
  - core
  - images
  - install
  - js
  - lib
  - payments
  - schemas
  - shippings
  - skins
    - basic
      - admin
      - customer
        - addons
        - blocks
        - buttons
        - common_templates
        - css
        - images
        - node_modules
        - pickers
        - views
        - base.css
        - base.styl
        - demo_skin_selector.tpl
        - design_mode.css
        - dropdown.css
        - dropdown.styl
        - exception.tpl
        - index.tpl
        - meta.tpl
        - print.css
        - styles.css
        - styles.styl
        - styles_ie.css
        - top_quick_links.tpl
        - variables.css
        - variables.styl

```
@import 'css/nib'
@import 'variables'

::selection
  background lighten(marablue, 80%)
  color darken(color-text, 80%)

// FONTS

font-yanone = 'Yanone Kaffeesatz', 'sans-serif'
font-cabin = 'Cabin Condensed', 'sans-serif'
font-ropa = 'Ropa Sans', 'sans-serif'
font-dosis = 'Dosis', 'sans-serif'


// general styles

body, h1, h2, h3, h4, h5, h6
  font normal 13px Arial,Tahoma,Helvetica,sans-serif

body, div, p
  color #333

.input-text, .input-text-auto, .input-text-large, .input-text-medium, .input-text-short, .input-text-100, .i
  background-color #fff
  color #2d2d2d
  vertical-align middle

input
  &[type="text"], &[type="password"]
    -webkit-box-sizing border-box
    -moz-box-sizing border-box
    box-sizing border-box
    margin 0 3px 0 0
    padding 6px 3px 5px
    border 1px solid #ccc
    font normal 100% Arial,Tahoma,Helvetica,sans-serif

textarea, select, .scroll-y
  -webkit-box-sizing border-box
  -moz-box-sizing border-box
  box-sizing border-box
  margin 0 3px 0 0
  padding 6px 3px 5px
  border 1px solid #ccc
  font normal 100% Arial,Tahoma,Helvetica,sans-serif

```

Terminal — tmux — 238×47

```
..
+ cache/
+ compile/
+ interface/
+ ivy/
+ launch/
+ licenses/
+ main/
+ project/
+ run/
+ sbt/
+ scripted/
+ src/
+ target/
+ tasks/
+ testing/
+ util/
  CONTRIBUTING.md
  LICENSE
  NOTICE
  README.md
  api.specification
  scripted.specification
```

```
62 >.def groupBy[K](discriminator: (File => K)): Map[K, Analysis]
63
64 >.override lazy val toString = Analysis.summary(this)
65 }
66
67 object Analysis
68 {
69 >.lazy val Empty: Analysis = new MAnalysis(Stamps.empty, APIs.empty, Relations.empty, SourceInfo
   s.empty, Compilations.empty)
70 >.private[sbt] def empty(nameHashing: Boolean): Analysis = new MAnalysis(Stamps.empty, APIs.empt
   y,
71 >.>.Relations.empty(nameHashing = nameHashing), SourceInfos.empty, Compilations.empty)
72
73 >./** Merge multiple analysis objects into one. Deps will be internalized as needed. */
74 >.def merge(analyses: Traversable[Analysis]): Analysis = {
75 >.>.if (analyses.exists(_.relations.nameHashing))
76 >.>.>.throw new IllegalArgumentException("Merging of Analyses that have" +
77 >.>.>.>."`relations.memberRefAndInheritanceDeps` set to `true` is not supported.")
78
79 >.>.// Merge the Relations, internalizing deps as needed.
80 >.>.val mergedSrcProd = Relation.merge(analyses map { _.relations.srcProd })
81 >.>.val mergedBinaryDep = Relation.merge(analyses map { _.relations.binaryDep })
82 >.>.val mergedClasses = Relation.merge(analyses map { _.relations.classes })
83
84 >.>.val stillInternal = Relation.merge(analyses map { _.relations.direct.internal })
85 >.>.val (internalized, stillExternal) = Relation.merge(analyses map { _.relations.direct.externa
   l }) partition { case (a, b) => mergedClasses._2s.contains(b) }
86 >.>.val internalizedFiles = Relation.reconstruct(internalized.forwardMap mapValues { _ flatMap m
   ergedClasses.reverse })
87 >.>.val mergedInternal = stillInternal ++ internalizedFiles
88
89 >.>.val stillInternalPI = Relation.merge(analyses map { _.relations.publicInherited.internal })
90 >.>.val (internalizedPI, stillExternalPI) = Relation.merge(analyses map { _.relations.publicInhe
   rited.external }) partition { case (a, b) => mergedClasses._2s.contains(b) }
91 >.>.val internalizedFilesPI = Relation.reconstruct(internalizedPI.forwardMap mapValues { _ flatM
   ap mergedClasses.reverse })
92 >.>.val mergedInternalPI = stillInternalPI ++ internalizedFilesPI
93
94 >.>.val mergedRelations = Relations.make(
95 >.>.>.mergedSrcProd,
96 >.>.>.mergedBinaryDep,
97 >.>.>.Relations.makeSource(mergedInternal, stillExternal),
98 >.>.>.Relations.makeSource(mergedInternalPI, stillExternalPI),
99 >.>.>.mergedClasses
```

```
887 [info] Resolving o
888 [info] Resolving o
889 [info] Resolving o
890 [info] Resolving o
891 [info] Resolving o
892 [info] Resolving o
893 [info] Resolving o
894 [info] Resolving o
895 [info] Resolving o
896 [info] Resolving o
897 [info] Resolving o
898 [info] Resolving o
899 [info] Resolving o
900 [info] Resolving c
901 [info] Resolving o
902 [info] Resolving o
903 [info] Resolving o
904 [info] Resolving o
905 [info] Resolving o
906 [info] Resolving o
907 [info] Resolving o
908 [info] Resolving o
909 [info] Resolving o
910 [info] Resolving o
911 [info] Resolving o
912 [info] Resolving o
913 [info] Resolving o
914 [info] Resolving o
915 [info] Resolving o
916 [info] Resolving o
917 [info] Resolving o
918 [info] Resolving o
919 [info] Done updati
920 [success] Total ti
921 root> ~compile
922 [success] Total ti
923 1. Waiting for sou
924
925 root> project main
926 [info] Set current
927 Main> ~compile
928 [success] Total ti
929 1. Waiting for sou
930
```

```
./vimfiler:explorer  2:1  NORMAL  0.13.5 > compile/inc/src/main/scala/sbt/inc/Analysis.scala  scala  25%  67:3  ./[vimshell] - shell
vim-memo  1 > vim  2 > vim
```

Write readable code

RStudio

diamondPricing.R | diamonds

```
1  library(ggplot2)
2
3  View(diamonds)
4  summary(diamonds)
5
6  summary(diamonds$price)
7  avgSize <- round(mean(diamonds$carat), 4)
8  clarity <- levels(diamonds$clarity)
9
10 q <- qplot(carat, price,
11            data=diamonds, color=clarity,
12            xlab="Carat", ylab="Price", main="Diamond Pricing")
13
14
```

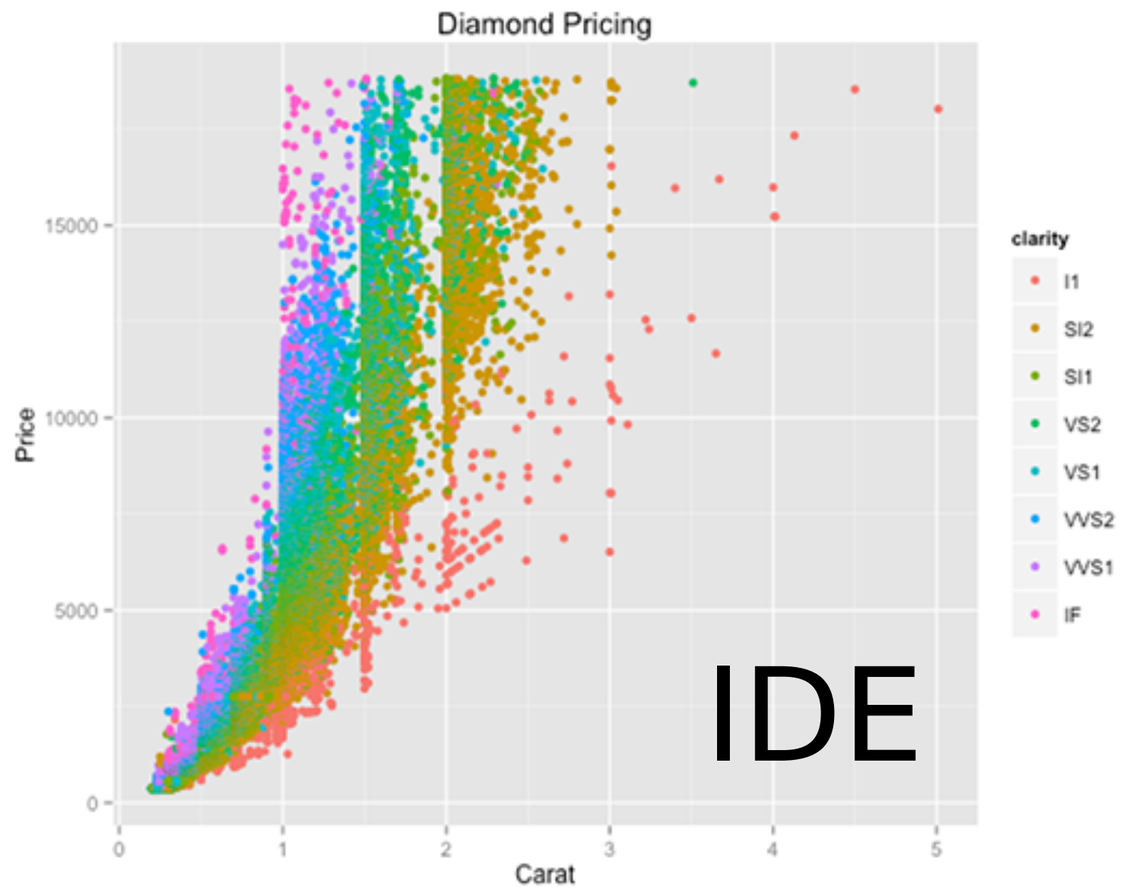
```
3rd Qu.:1.0400   Ideal     :21551   H: 8304   VVS2    : 5066   3rd Qu.:62.50
Max.   :5.0100                      I: 5422   VVS1    : 3655   Max.   :79.00
                                    J: 2808   (Other): 2531
     table           price             x                y                z
 Min.   :43.00   Min.   :  326   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
 1st Qu.:56.00   1st Qu.:  950   1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910
 Median :57.00   Median : 2401   Median : 5.700   Median : 5.710   Median : 3.530
 Mean   :57.46   Mean   : 3933   Mean   : 5.731   Mean   : 5.735   Mean   : 3.539
 3rd Qu.:59.00   3rd Qu.: 5324   3rd Qu.: 6.540   3rd Qu.: 6.540   3rd Qu.: 4.040
 Max.   :95.00   Max.   :18823   Max.   :10.740   Max.   :58.900   Max.   :31.800

> summary(diamonds$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    326     950    2401    3933    5324   18820
> avgSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> q <- qplot(carat, price,
+            data=diamonds, color=clarity,
+            xlab="Carat", ylab="Price", main="Diamond Pricing")
> q
>
```

Environment | History

Global Environment

Data	
diamonds	53940 obs. of 10 variables
Values	
avgSize	0.7979
clarity	chr [1:8] "I1" "SI2" "SI1" "VS2" "VS1" "VVS2" "VVS1" "…
q	Large gg (9 elements, 3.3 Mb)

Files | Plots | Packages | Help | Viewer



Write readable code

Rollply: inahull_cpp.cpp — Kate

File Edit View Projects Bookmarks Sessions Tools Settings Help

Projects

```r
#'
#' @title Build a regular grid
#'
#' @description Create a regular grid with the same number of points on each
#'              dimension.
#'
#' @param coords A matrix or data frame of coordinates as columns
#' @param npts The approximate total number of points of the output grid
#' @param pad Padding on each dimension (a positive number makes a grid
#'            that is larger than the ranges of the coordinates).
#' @param ... other arguments are silently ignored
#'
#' @return The coordinates of a grid of points as a data frame with
#'         approximately \code{npts} rows and \code{ncol(coords)} columns. Names
#'         are transfered from the \code{coords} data frame.
#'
#' @details This function creates a grid that covers a set of points. The
#'          number of points of the output grid is the same on all dimensions.
#'          This is probably the only useful option for 1D moving-window
#'          computations.
#'
#' @family grid building functions
#'
#' @export
#'

build_grid_identical <- function(coords, npts, pad = 0, ...) {

  build_grid_check_vars(coords, npts)

  coords.ranges <- apply(coords, 2, range)
  ndims <- ncol(coords)

  # The input npts is the *total* number of points wanted so we need ^(1/ndims)
  # so we use the correct, identical amount of points on each dimension.
  length.onedim <- floor(npts^(1/ndims))
```

```cpp
//
//
// This function is an alternative to alphahull::inahull designed
//   for speed.
//

#include <Rcpp.h>
#include <stdlib.h>

// Define column names of $complement in hull object
#define COMP_C1 0 //
#define COMP_C2 1 //
#define COMP_R 2 // third column in complement matrix (with indexing difference)

// Relaxed floating-point equality
#define feq(x,y) ( std::fabs(x - y) < 1e-10 ? true : false)

using namespace Rcpp;

// This is a c++ implementation of alphahull::inahull
//[[Rcpp::export]]
LogicalVector inahull_cpp_multiple(List ahull_obj,
                                   NumericVector X,
                                   NumericVector Y) {

  NumericMatrix comp = as<NumericMatrix>(ahull_obj["complement"]);
  LogicalVector inhull = LogicalVector(X.length());



  // Get list of balls/halfplanes
  NumericVector halfpl = NumericVector(comp.nrow());
  NumericVector ball   = NumericVector(comp.nrow());
  int n_halfpl = 0;
  int n_ball = 0;
  for (int i=0; i<comp.nrow(); i++) {
```

Documents Snippets

```
Error: No tests found for .
In addition: Warning message:
In library(package, lib.loc = lib.loc, character.only = TRUE, logical.return = TRUE,  :
  there is no package called '.'
> library(devtools)
> test()
Loading rollply
Testing rollply
Handling of bad arguments : .....
Grid building: grid_identical : ..
Testing inahull_cpp : .
Common uses of rollply : ..

DONE
> 
```

Search and Replace | Terminal | Current Project

Write readable code

Write for people first ! (*you included*)

Write with *style !*

Google's style guide

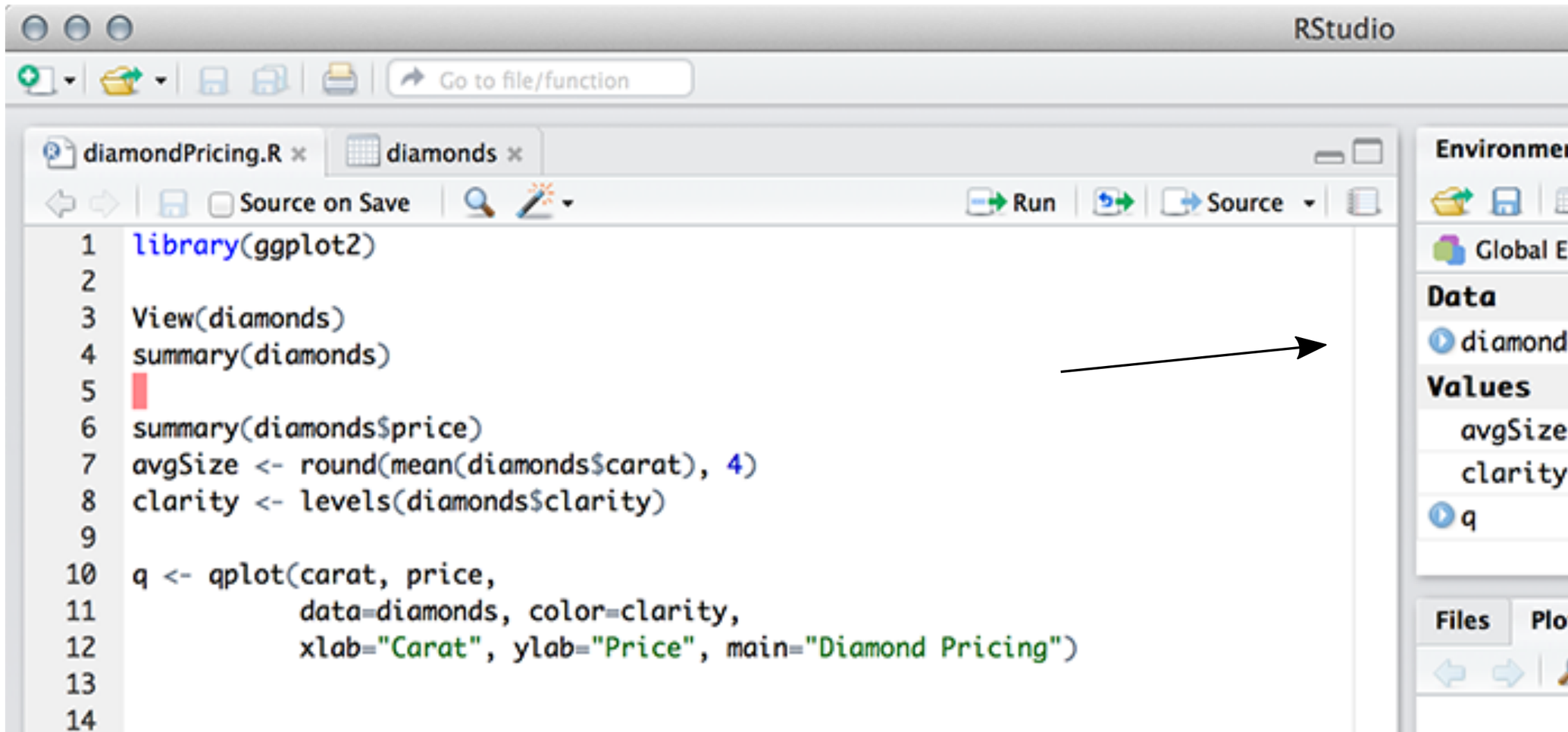
Hadley's style guide

Write readable code

Write for people first ! (*you included*)

Write with *style !*

Do not write your whole program in a single line that takes most of my fuck



Write readable code

Write for people first ! (*you included*)

Write with *style !*

Do not write your whole program in a single line that takes most of my fuck

Comments

```
i <- i + 1 # add one to i
```

```
tmp <- head(iris) # store the first few lines of iris (a df)
```

Write readable code

Write for people first ! (*you included*)

Write with *style !*

Do not write your whole program in a single line that takes most of my fuck

Comments

```
i <- i + 1 # add one to i
```

```
tmp <- head(iris) # store the first few lines of iris (a df)
```

Adapt to your public: *who is reading your code ?*

when teaching

when working with colleagues

Write readable code

ggplot()

ode() # from pkg deSolve

%>% # from pkg magrittr

```
# from unknown package
f(1) <- 1
f(n) <- f(n) + f(n-1)
```

sim() # from package simecol

Write readable code

Write for people first ! (*you included*)

Write with *style !*

Do not write your whole program in a single line that takes most of my fuck

Comments

```
i <- i + 1 # add one to i
```

```
tmp <- head(iris) # store the first few lines of iris (a df)
```

Adapt to your public: *who is reading your code ?*

when teaching

when working with colleagues

Be expressive

Write readable code

Good names

Variables

One letter

```
i,j,k, etc.
x, y
```

A few letters

```
tmp
dat, mat
df
```

Many letters

```
plot_with_x_axis_off
df_without_site_21240
```

Functions

Think of verbs !
Functions are actions

```
add(1, to = 3)
```

```
simu_set_tstop(time = 3000)
stop_simu(at = 3000)
```

Write readable code

Good names

Variables

One letter

```
i,j,k, etc.
x, y
```

A few letters

```
tmp
dat, mat
df
```

Many letters

```
plot_with_x_axis_off
df_without_site_21240
```

Functions

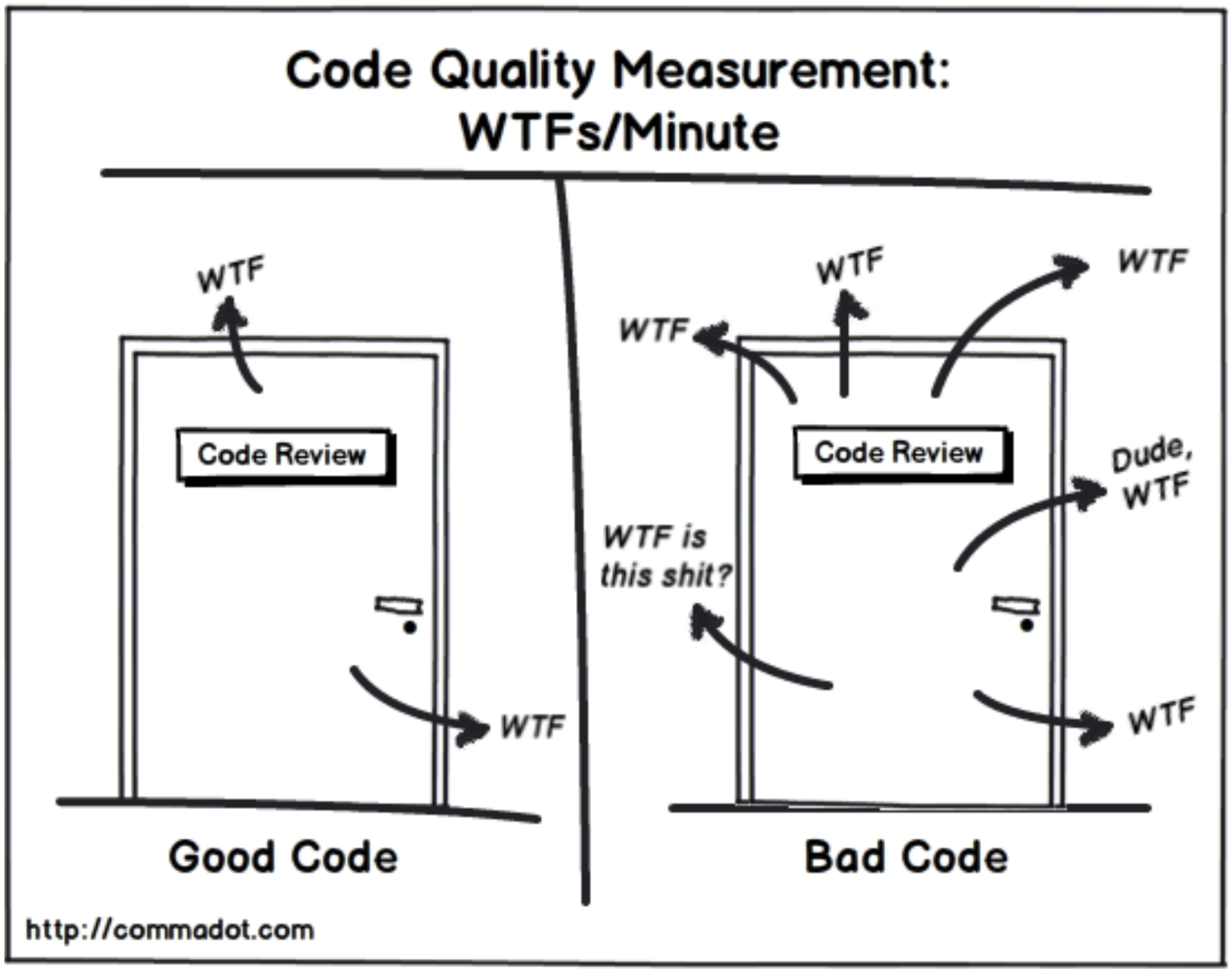
Think of verbs !
Functions are actions

```
add(1, to = 3)
```

```
simu_set_tstop(time = 3000)
stop_simu(at = 3000)
```

Write readable code

Your turn: code review



Checkpoint

Use the right programming paradigm

Recognize analysis patterns

Write readable code

Choose the right data structures

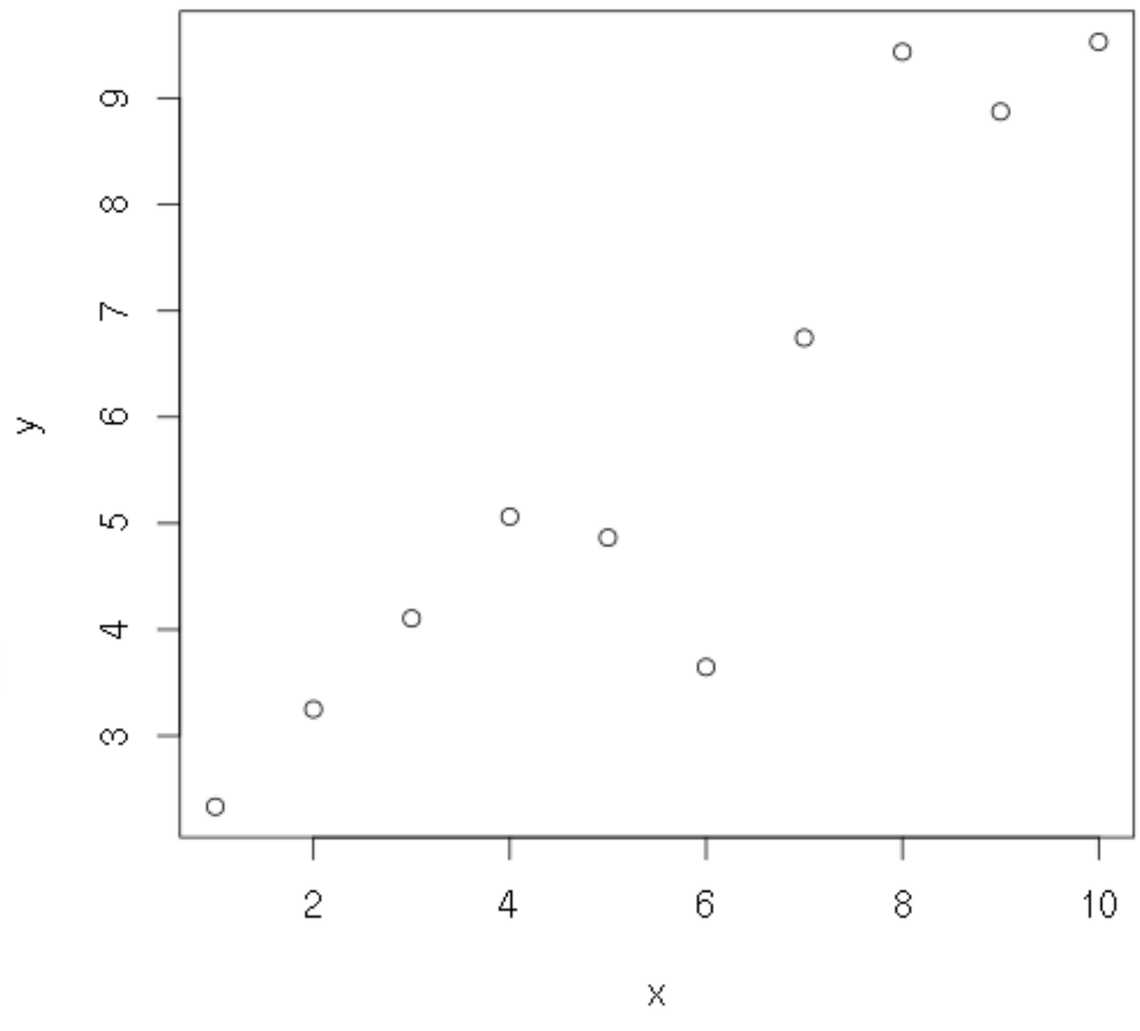
Organize your workspace

Patterns in analyses

Break it down into pieces: FUNCTIONS
Why ? DRY. Separation of concerns. Testing.

```r
# Get the closest value in a vector
closest_to <- function(val, X, quiet = FALSE) {
  index <- which( abs(val - X) == min(abs(val - X)) )
  new_value <- X[min(index)]
  if (!quiet) {
    cat('Picked value ', new_value, ' (error:
  }
  return(new_value)
}
```

```
> x <- seq.int(10)
> y <- x + rnorm(10)
> plot(x,y)
> closest_to(6, y)
Picked value 6.745802 (error: 0.745
[1] 6.745802
>
```

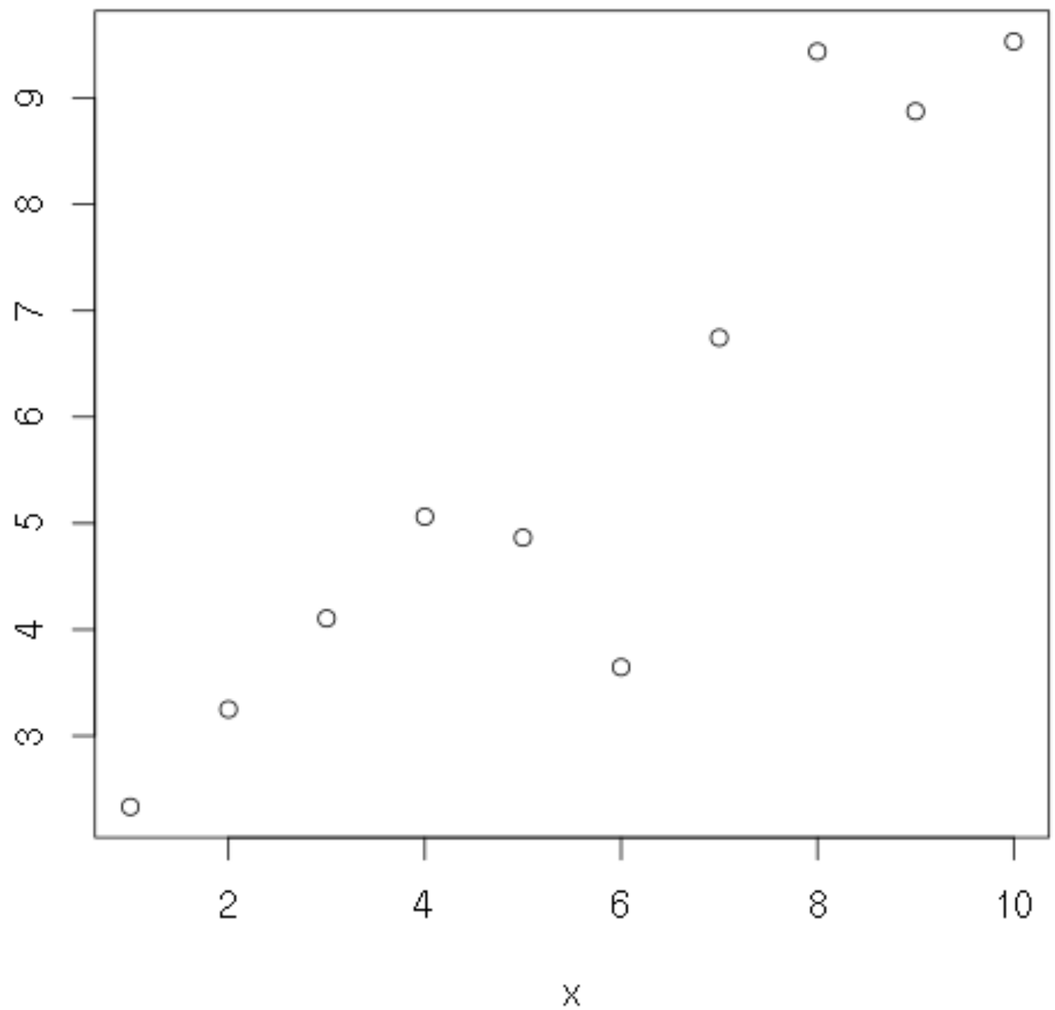


Patterns in analyses

Break it down into pieces: FUNCTION
Why ? DRY. Separation of concerns. T

```r
# Get the closest value in a vector
closest_to <- function(val, X, quiet = FALSE) {
  index <- which( abs(val - X) == min(abs(val -
  new_value <- X[min(index)]
  if (!quiet) {
    cat('Picked value ', new_value, ' (error:
  }
  return(new_value)
}
```

```
> x <- seq.int(10)
> y <- x + rnorm(10)
> plot(x,y)
> closest_to(6, y)
Picked value 6.745802 (error: 0.745802)
[1] 6.745802
>
```



Patterns in analyses

```r
# Print the error between an expected value and what is observed
print_error <- function(expected, observed) {
  print0('Picked value ', observed, ' (error: ', observed - expected,')')
}

# Are the values the minima of a vector ?
# e.g. for c(1, 2, 1), this function returns c(TRUE, FALSE, TRUE)
is_minimum <- function(vec) {
  vec == min(vec)
}

# Get the closest value in a vector
closest_to <- function(val, X, quiet = FALSE) {
  index <- which( is_minimum(abs(X-val)) )
  new_value <- X[min(index)]

  if ( !quiet ) {
    print_error(val, new_value)
  }

  return(new_value)
}
```

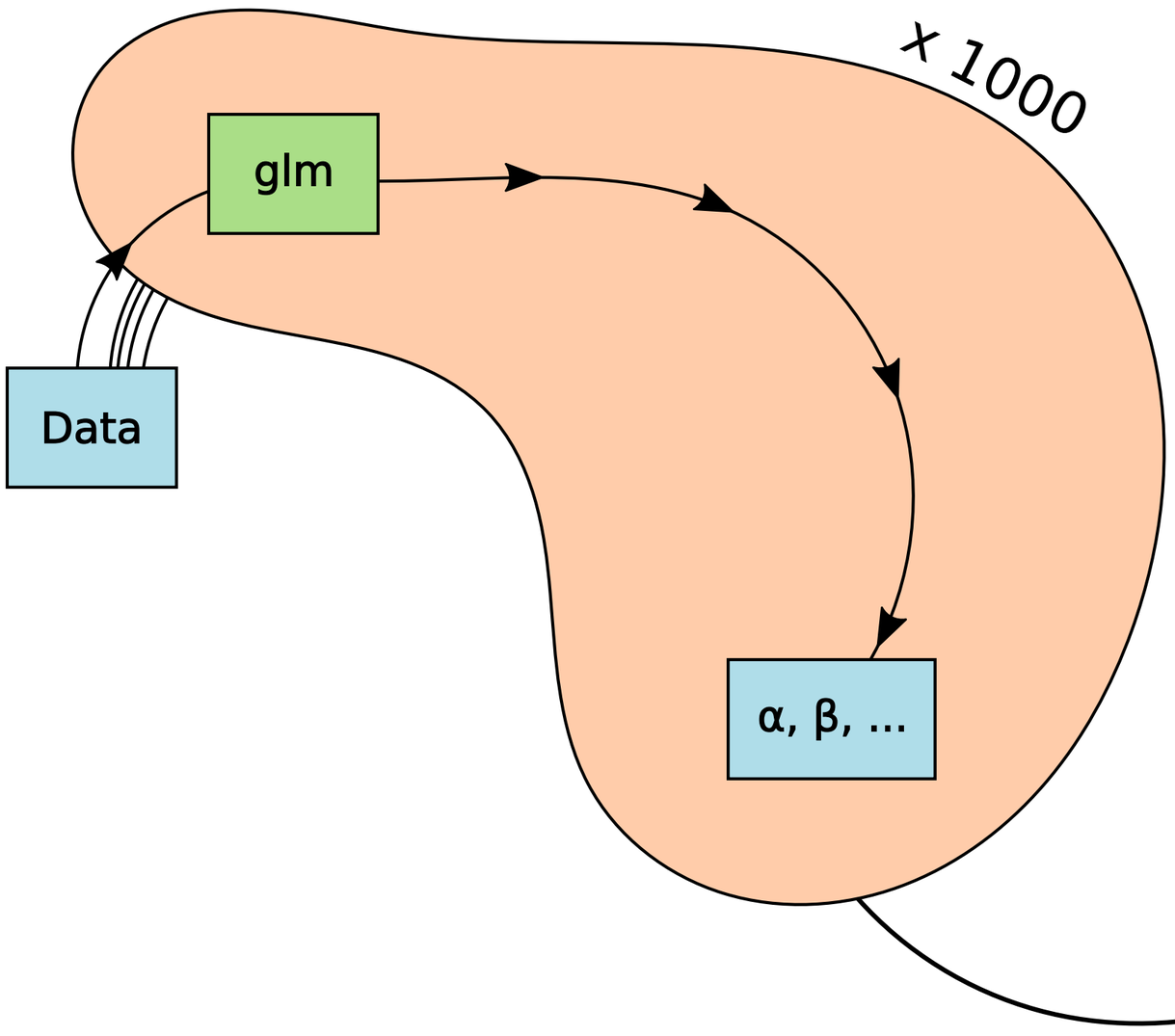
More expressive

Describe the what, not the how.
Dirty job is under the rug

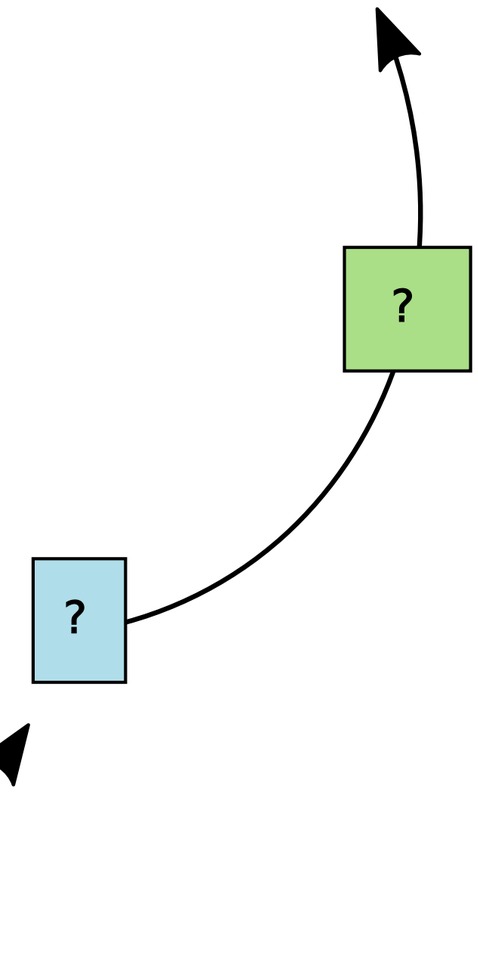
Patterns in analyses

Patterns in analyses

The Repeat-combine pattern

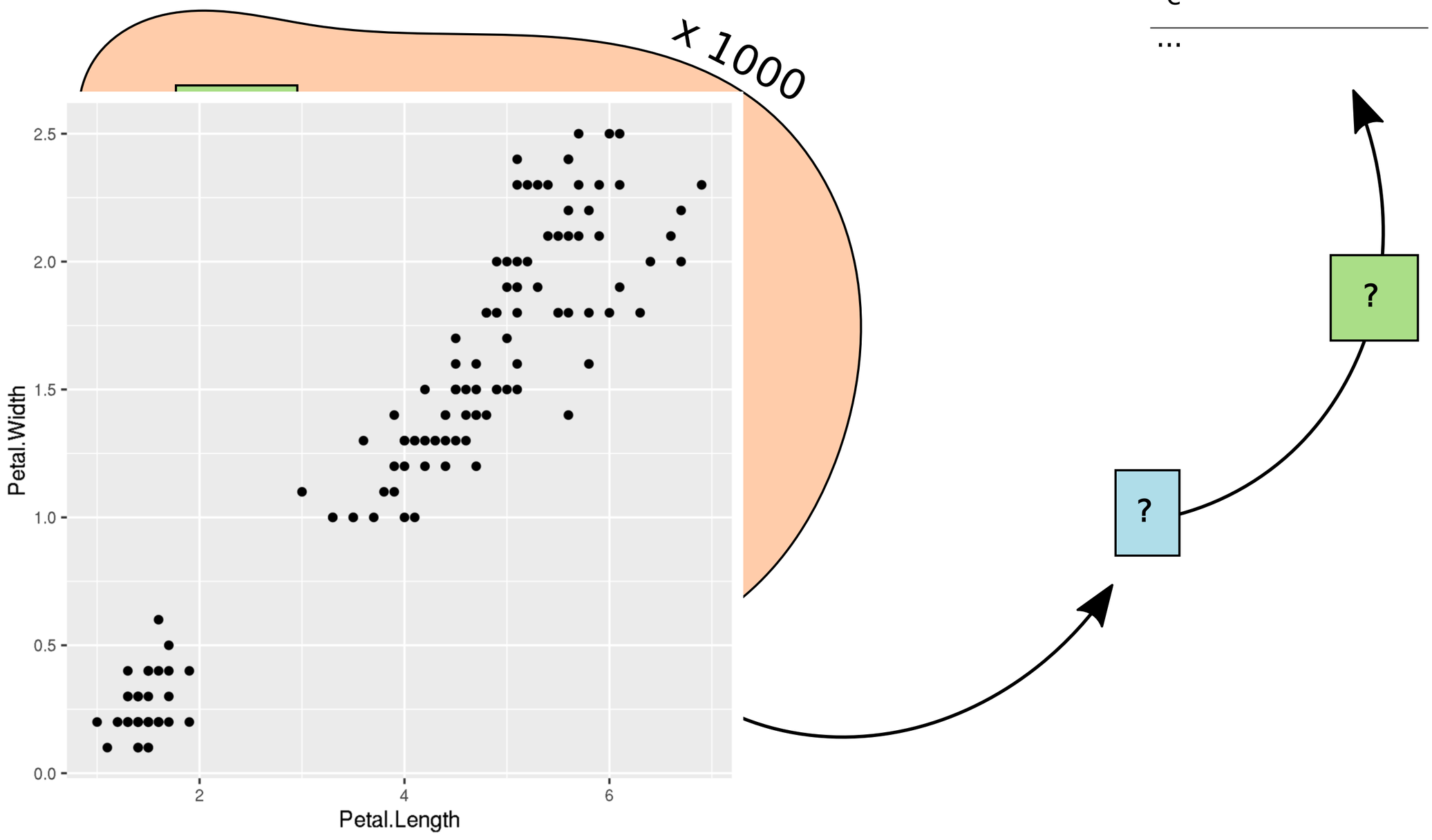


	Mean	SD
a		
b		
c		
...		



Patterns in analyses

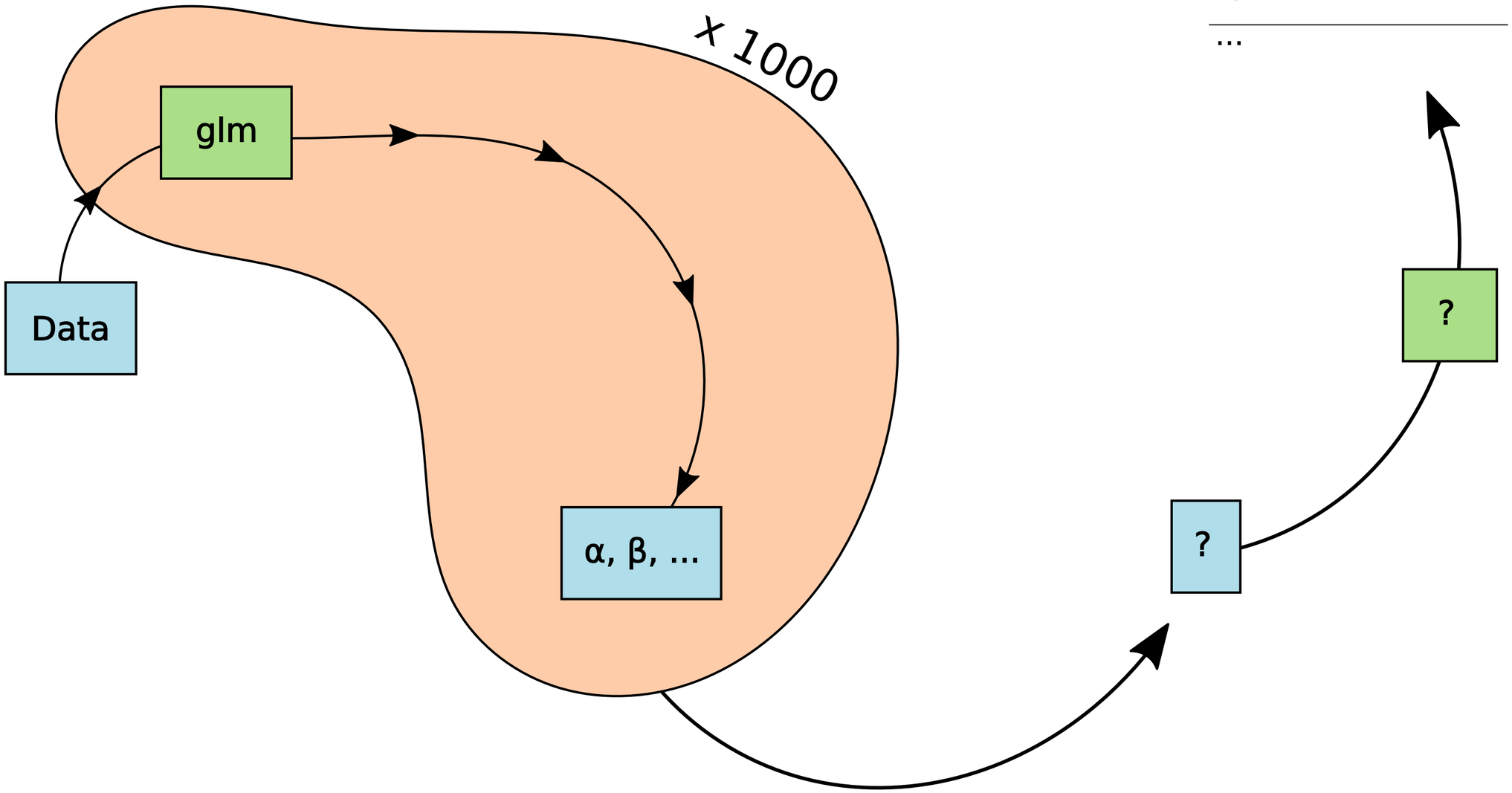
The Repeat-combine pattern



	Mean	SD
a		
b		
c		
...		

Patterns in analyses

The Repeat-combine pattern



	Mean	SD
a		
b		
c		
…		

Patterns in analyses

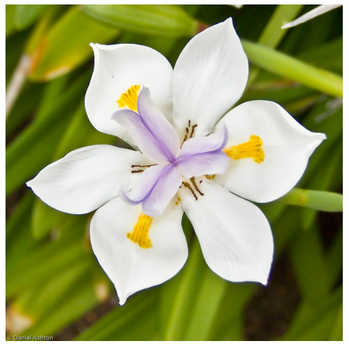
The Split - apply - combine pattern

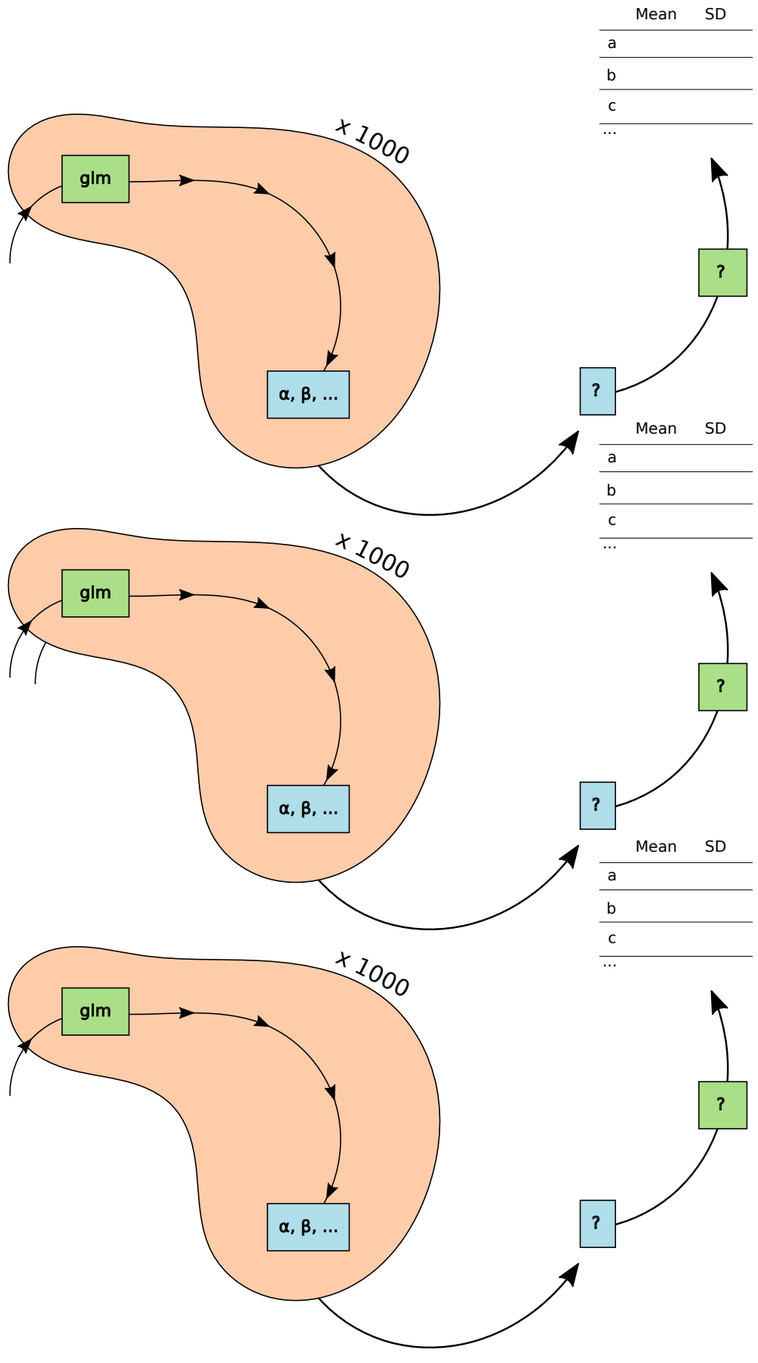
Data

Patterns in analyses

The Split - apply - combine pattern

Data

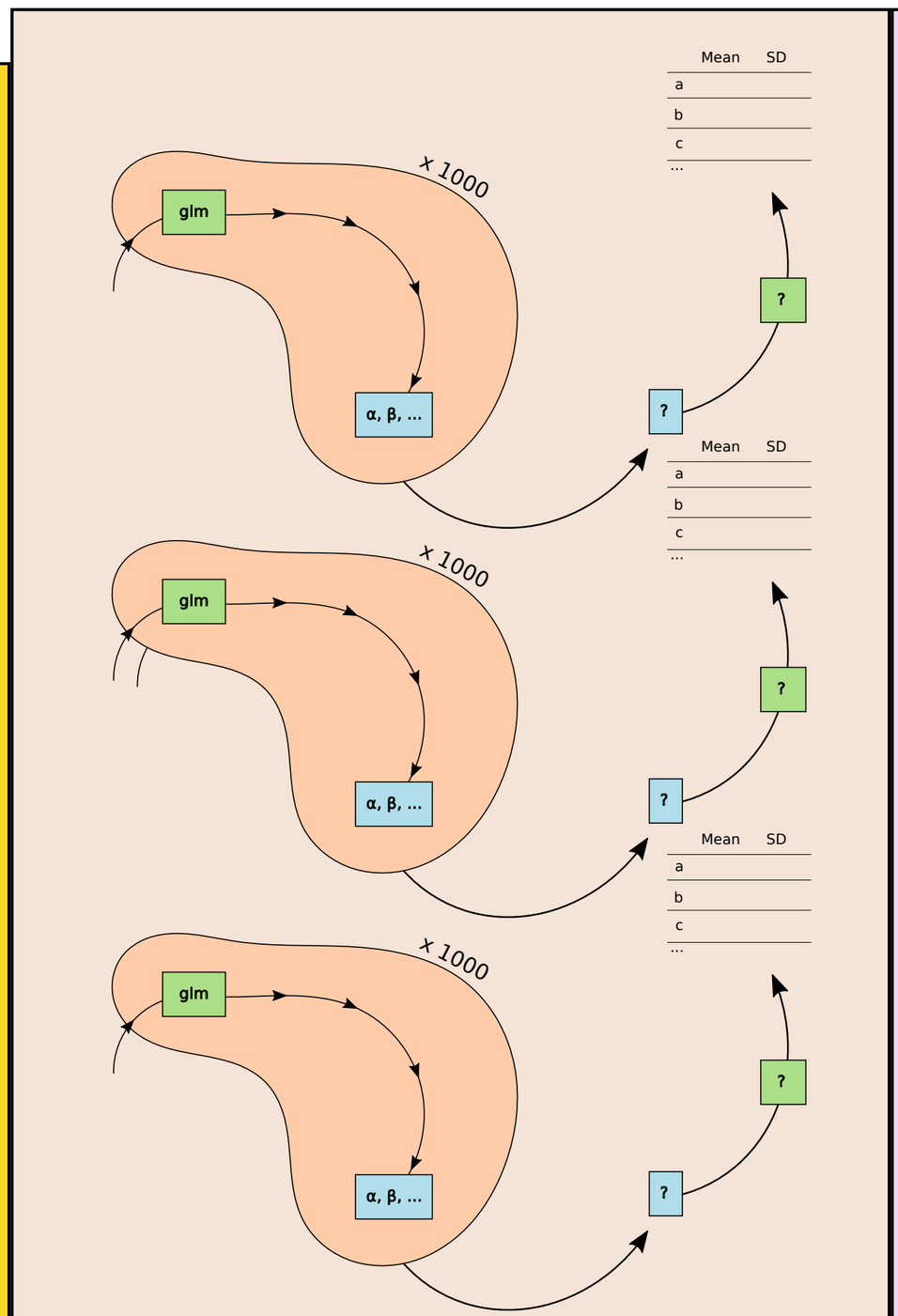




Patterns in analyses

The Split - apply - combine pattern

Data

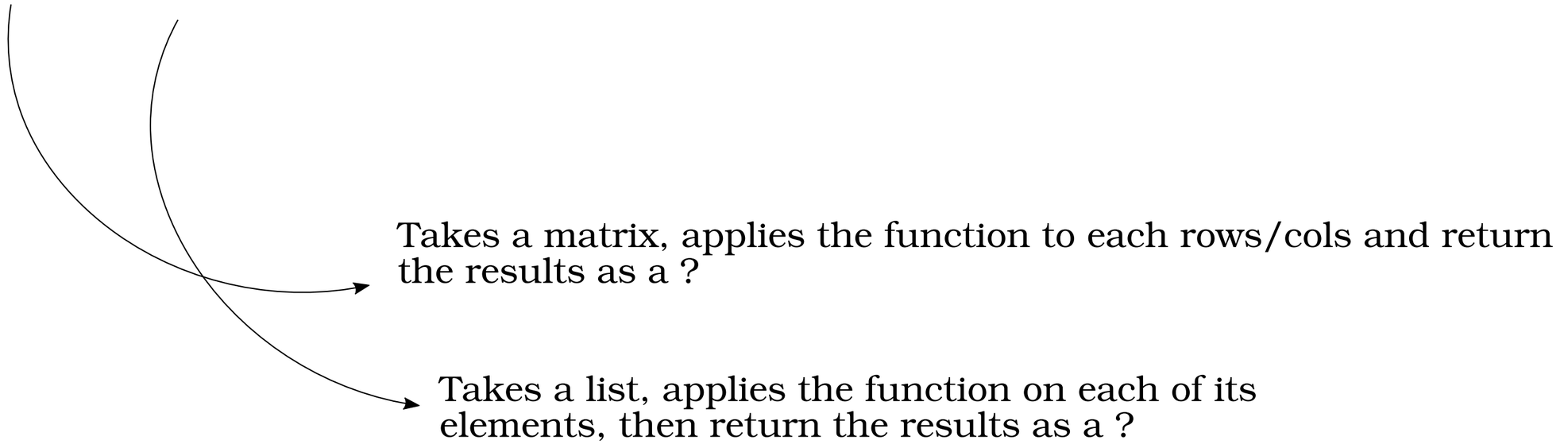


	Species	Mean	SD
a	setosa		
b	setosa		
c	virginica		
...			

Patterns in analyses

The split-apply-combine pattern

`apply/lapply/sapply/mapply/rapply/etc.`

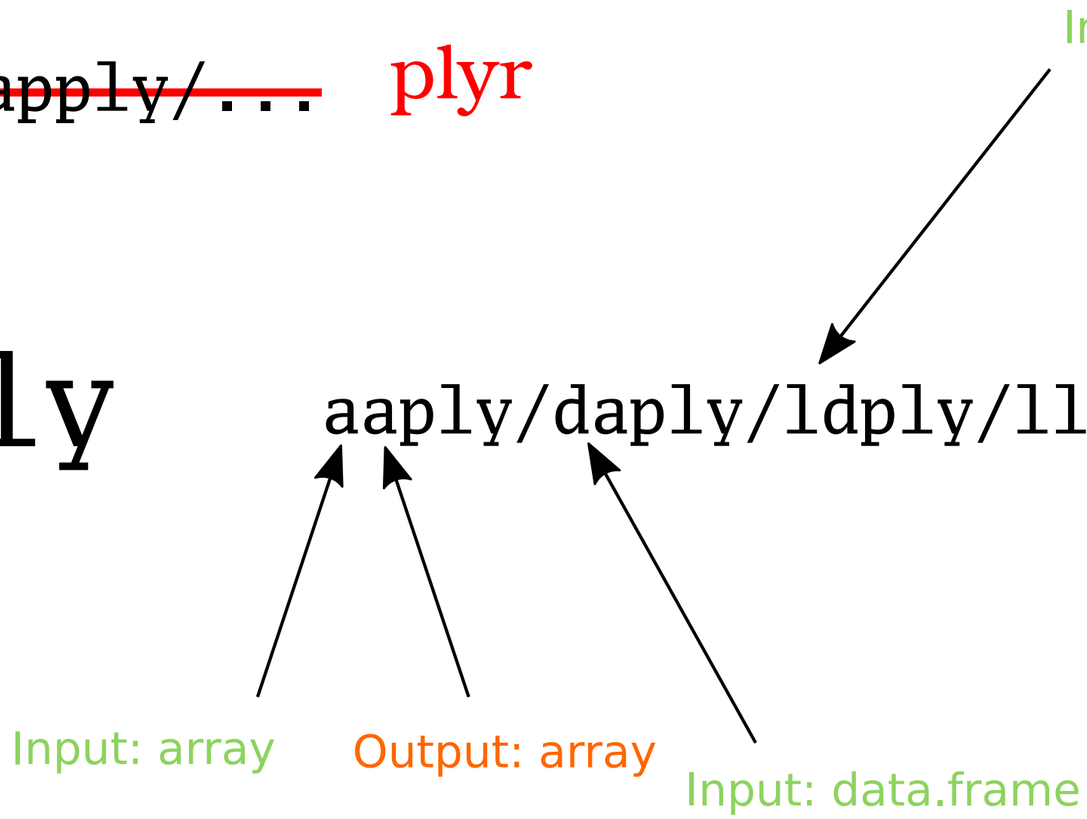


Patterns in analyses

The split-apply-combine pattern

~~apply/lapply/sapply/...~~ plyr

xyply



Patterns in analyses

The split-apply-combine pattern

```
ddply(iris, ~ Species,
      function(df) {
        data.frame(Sepal.Length.mean = mean(df[ ,'Sepal.Length']))
      })
```

```
ddply(iris, ~ Species,
      summarize, Sepal.Length.mean = mean(Sepal.Length)))
```

Patterns in analyses

The split-apply-combine pattern

```
ddply(iris, ~ Species,
      function(df) {
        data.frame(Sepal.Length.mean = mean(df[ ,'Sepal.Length']))
      })
```

```
ddply(iris, ~ Species,
      summarize, Sepal.Length.mean = mean(Sepal.Length)))
```

```
dlply(iris, ~ Species,
      function(split_df) {
        glm(Petal.Length ~ Petal.Width, data = dplit_df)
      })
```

Patterns in analyses

The split-apply-combine pattern

Compute the petal mean and SD of each iris species (dataset: iris)

Compute the mean price of diamonds by cut and color (dataset: diamonds)

Compute the bootstraped slope/intercept estimate for each iris species

Patterns in analyses

The split-apply-combine pattern

```
ddply(iris, ~ Species,
      function(df) {
        data.frame(Sepal.Length.mean = mean(df[ ,'Sepal.Length']))
      })
```

Patterns in analyses

Other analyses patterns ?

Checkpoint

Use the right programming paradigm

Recognize analysis patterns

Write readable code

Choose the right data structures

Organize your workspace

OOP basics: class, methods

A set of points ?

```
a <- c(06, 30, 15, 67, 12, 43)
```

```
print(a) ?
```

OOP basics: class, methods

A set of points ?

```
a <- c(06, 30, 15, 67, 12, 43)
```

```
print(a) ?
```

```
a <- c(06, 30, 15, 67, 12, 43)
```


```
print.telephone_number <- function(tel) {
  paste(c('Phone number:', a), collapse = ' ')
}
```

```
print(a) ?
```

OOP basics: class, methods

```
A set of points ?

    a <- c(06, 30, 15, 67, 12, 43)

       print(a) ?
```

polymorphism !

```
a <- c(06, 30, 15, 67, 12, 43)
```

telephone_number

method

```
print.telephone_number <- function(tel) {
  paste(c('Phone number:', a), collapse = ' ')
}
```

```
print(a) ?
```

OOP basics: class, methods

Polymorphism of the plot function

```
plot(iris)
plot(Sepal.Length ~ Sepal.Width, data = iris)
plot(list(x = seq.int(10),
          y = seq.int(10) + rnorm(10)))
```

OOP basics: class, methods

Polymorphism of the plot function

```
plot(iris)
plot(Sepal.Length ~ Sepal.Width, data = iris)
plot(list(x = seq.int(10),
          y = seq.int(10) + rnorm(10)))
```

Other functions ?

```
print.data.frame <- function(df) {
  print( summary(df) )
  cat(paste0('... (', nrow(iris), ' rows)\n'))
}
```

functional programming

OOP: insist on *what objects are*

functional programming: insists on *how we transform* objects