Slides/code on GitHub!

Algebraic Statistics in R

A State of the Union

David J. Kahle

Associate Professor

BAYLOR UNIVERSITY

1845

DEPARTMENT OF STATISTICAL SCIENCE

Overview

- Historical Perspective
- The **algstat** Ecosystem
  - Installing the packages
  - mpoly
  - latter
  - m2r
  - bertini
  - tropical
  - algstat
- GitHub and Contributing
- Upcoming Projects

Historical Perspective

Historical perspective

Base R has essentially no support for symbolic computing

Historical perspective

Base R has essentially no support for symbolic computing

But it does have:

Historical perspective

Base R has essentially no support for symbolic computing

But it does have:

a rich collection of base object types (S3, S4, R6, …)

Historical perspective

Base R has essentially no support for symbolic computing

But it does have:

- a rich collection of base object types (S3, S4, R6, …)
- a variety of ways to create new objects

Historical perspective

Base R has essentially no support for symbolic computing

But it does have:

a rich collection of base object types (S3, S4, R6, …)

a variety of ways to create new objects

flexible mechanisms for implementing new methods

(including infix operators and overloading)

Historical perspective

Base R has essentially no support for symbolic computing

But it does have:

- a rich collection of base object types (S3, S4, R6, …)
- a variety of ways to create new objects
- flexible mechanisms for implementing new methods (including infix operators and overloading)
- access to the operating system (rw, sockets, …)

Historical perspective

Base R has essentially no support for symbolic computing

But it does have:

a rich collection of base object types (S3, S4, R6, …)

a variety of ways to create new objects

flexible mechanisms for implementing new methods (including infix operators and overloading)

access to the operating system (rw, sockets, …)

a simple way to incorporate C++ routines (Rcpp)

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

mpoly

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis

mpoly

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

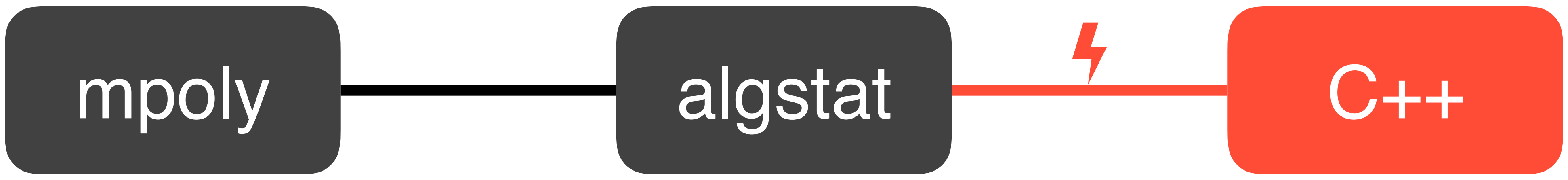
2014 : **algstat** – algebraic statistical data analysis

mpoly
algstat

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis

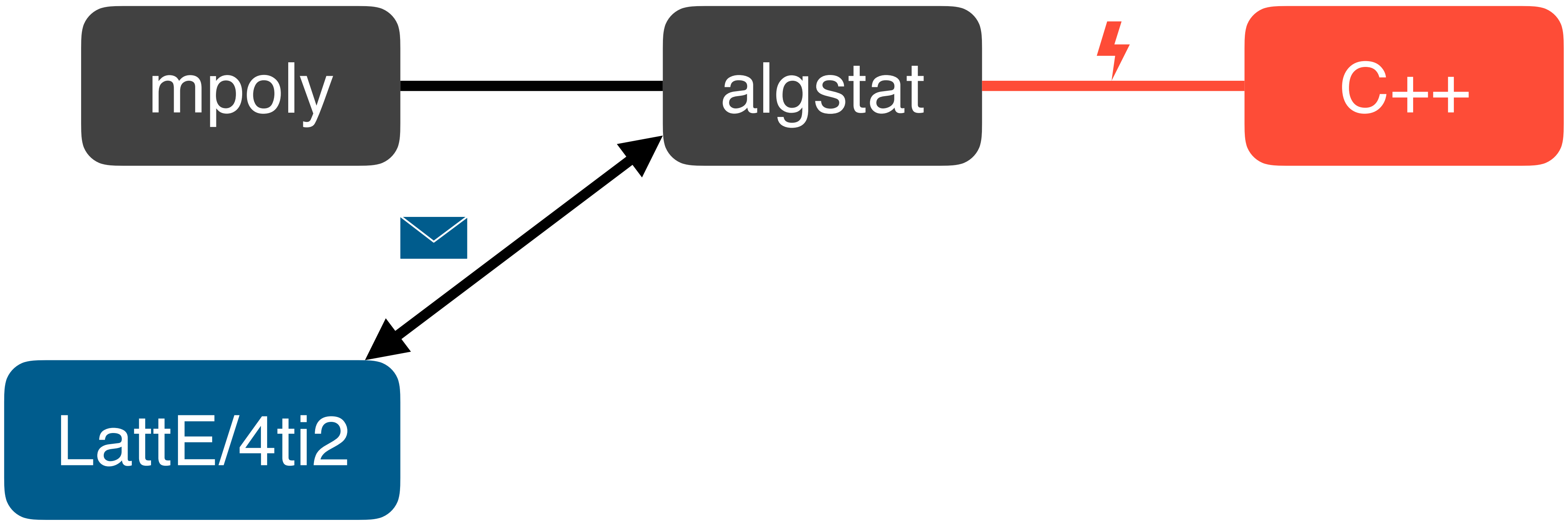


⚡ Objects computed on in place

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis



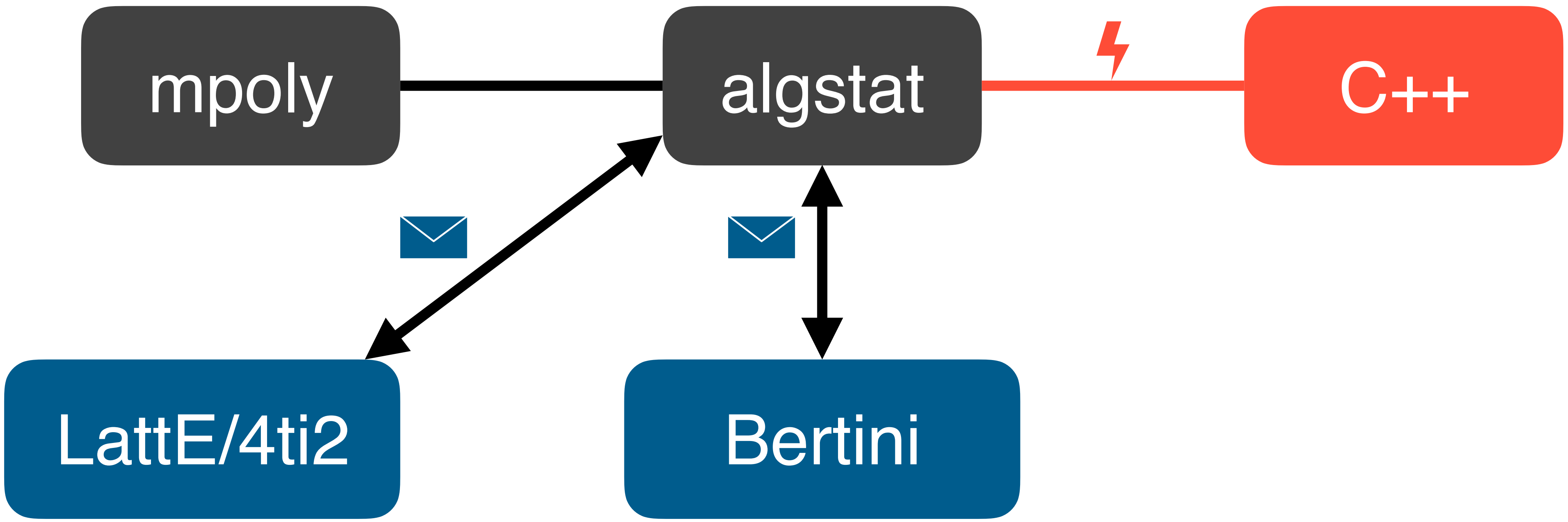
⚡ Objects computed on in place

✉ R writes / program executes / R reads

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis



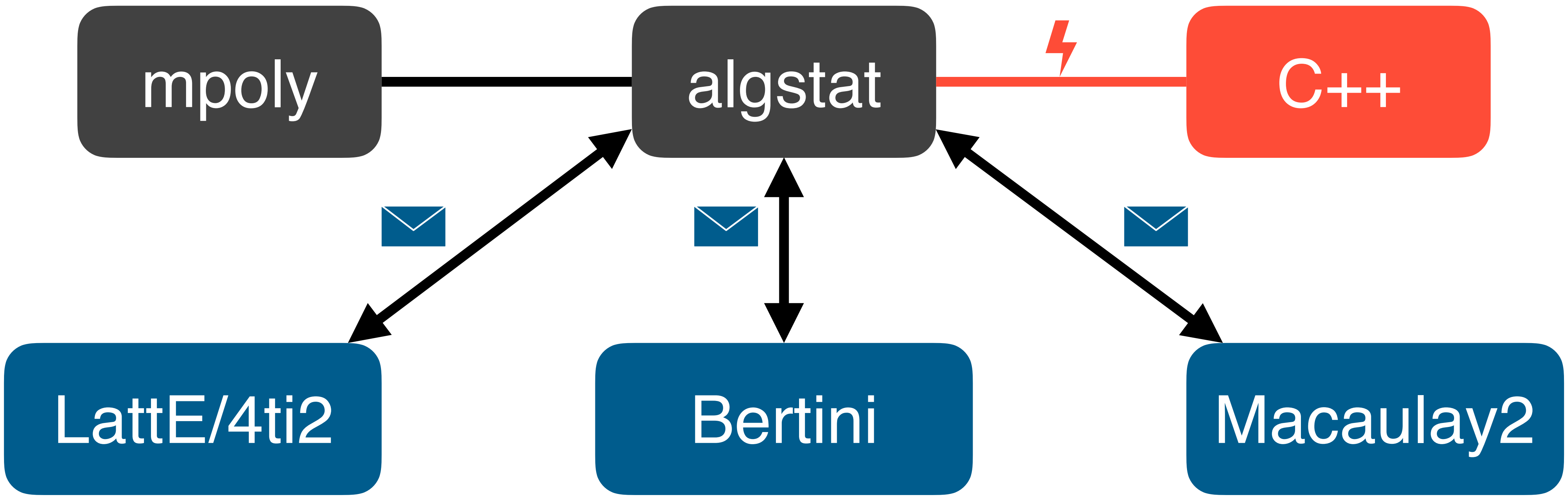
⚡ Objects computed on in place

✉ R writes / program executes / R reads

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis



⚡ Objects computed on in place

✉ R writes / program executes / R reads

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis

Then, connections needed their own packages!

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis

Then, connections needed their own packages!

2015 : **latter** – LattE/4ti2

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis

Then, connections needed their own packages!

2015 : **latter** – LattE/4ti2

2016 : **m2r** – Macaulay2

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis

Then, connections needed their own packages!

2015 : **latter** – LattE/4ti2

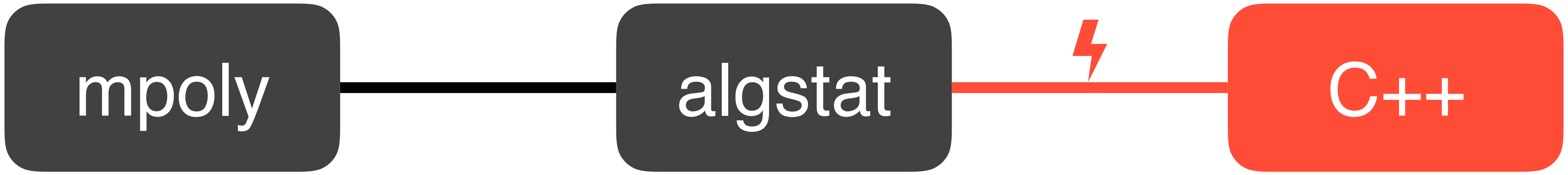
2016 : **m2r** – Macaulay2

2017 : **tropical** – Tropical geometry

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

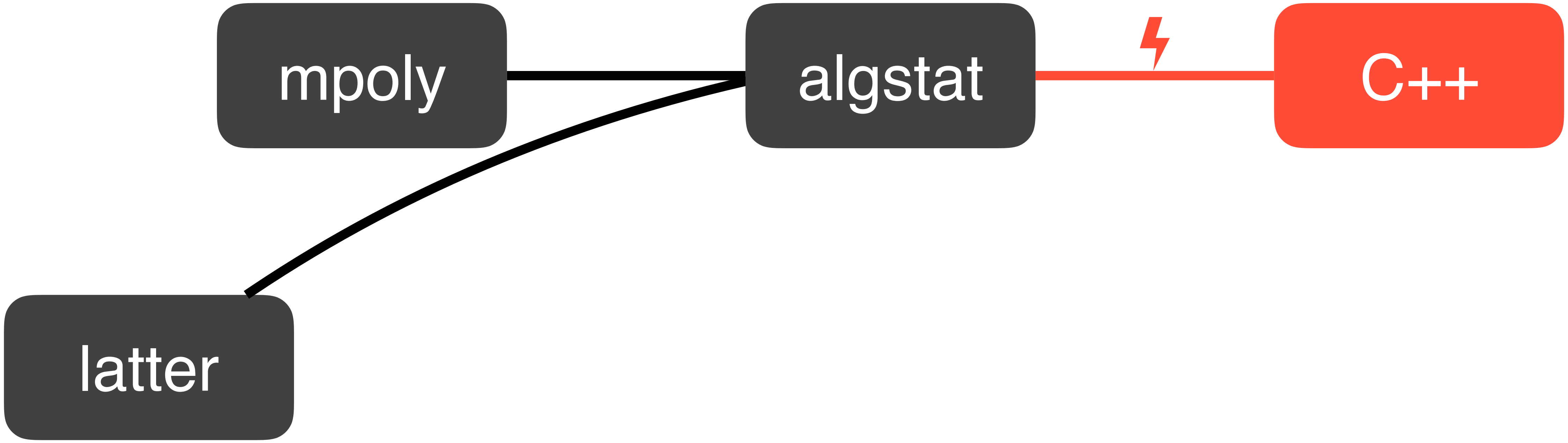
2014 : **algstat** – algebraic statistical data analysis



Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

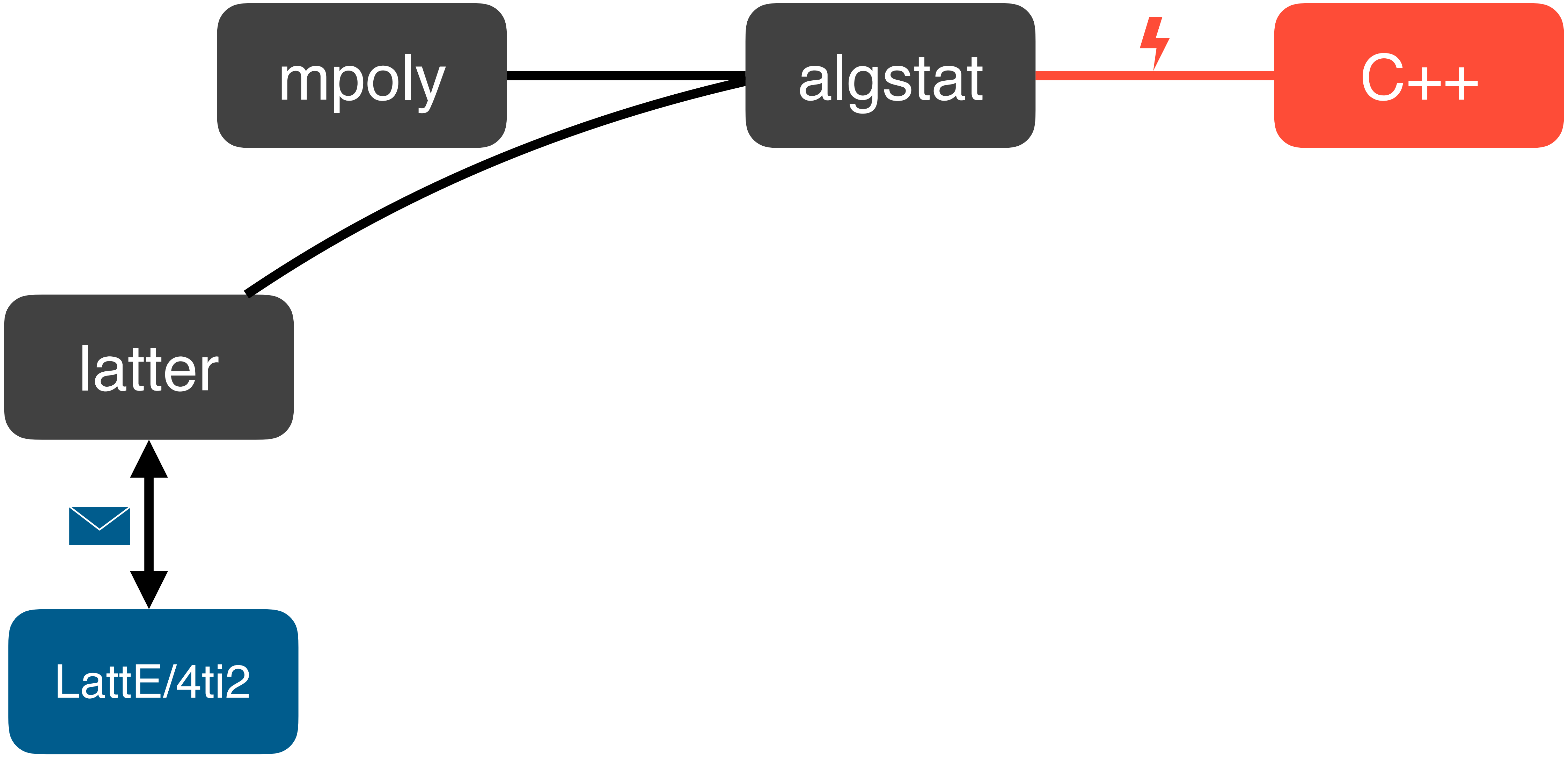
2014 : **algstat** – algebraic statistical data analysis



Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

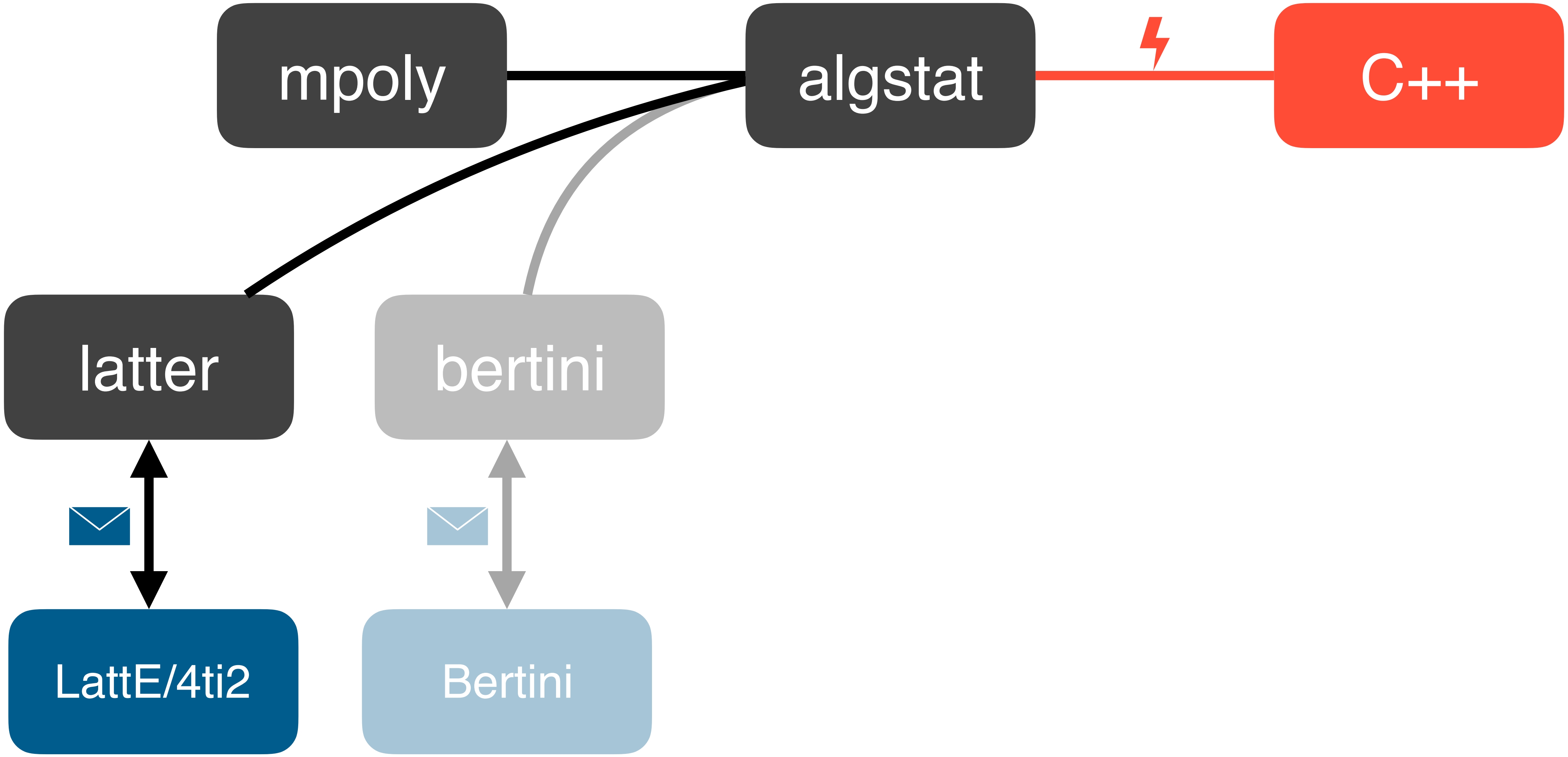
2014 : **algstat** – algebraic statistical data analysis



Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis



mpoly

algstat

C++

latter

bertini

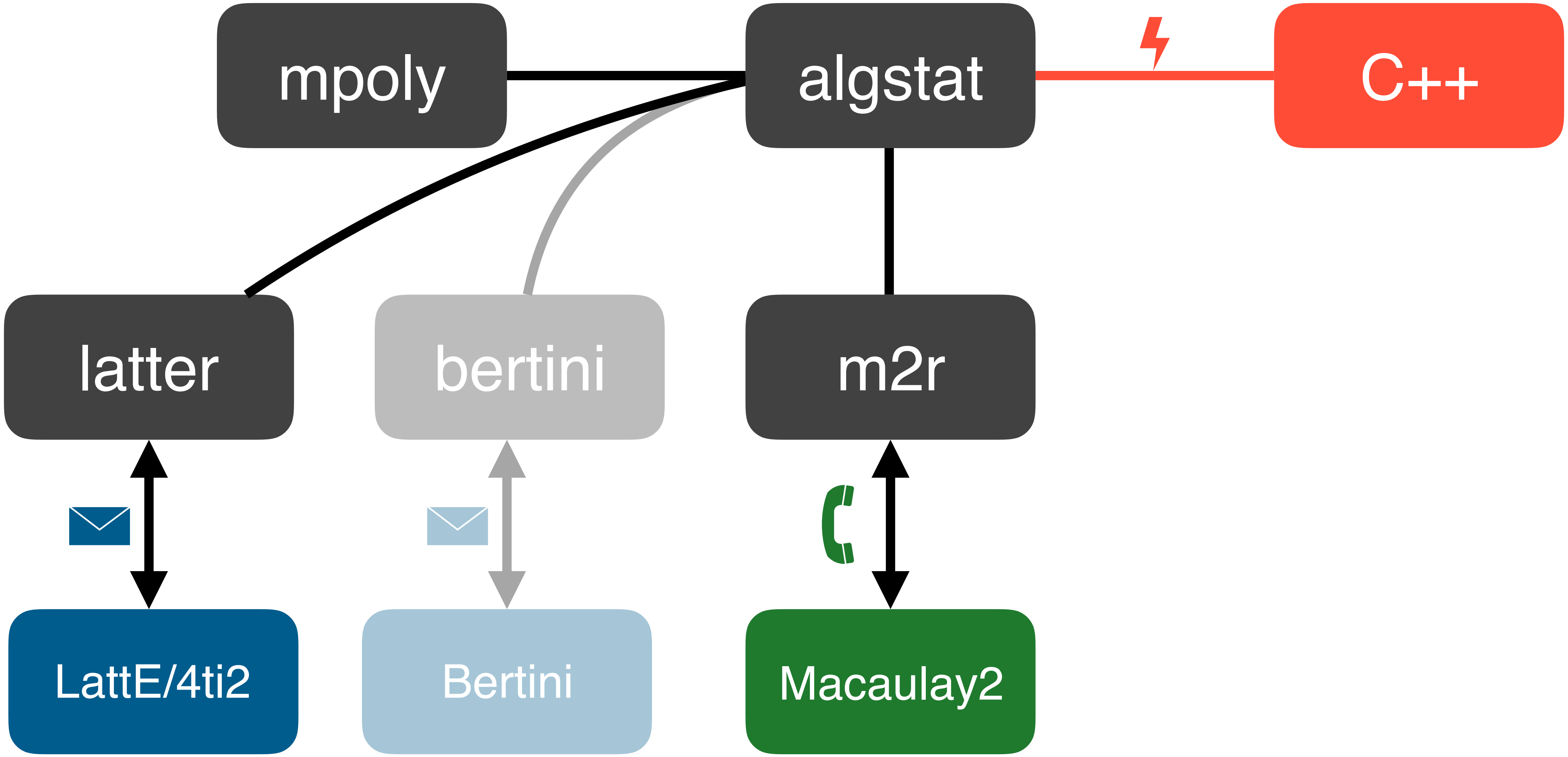
LattE/4ti2

Bertini

Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

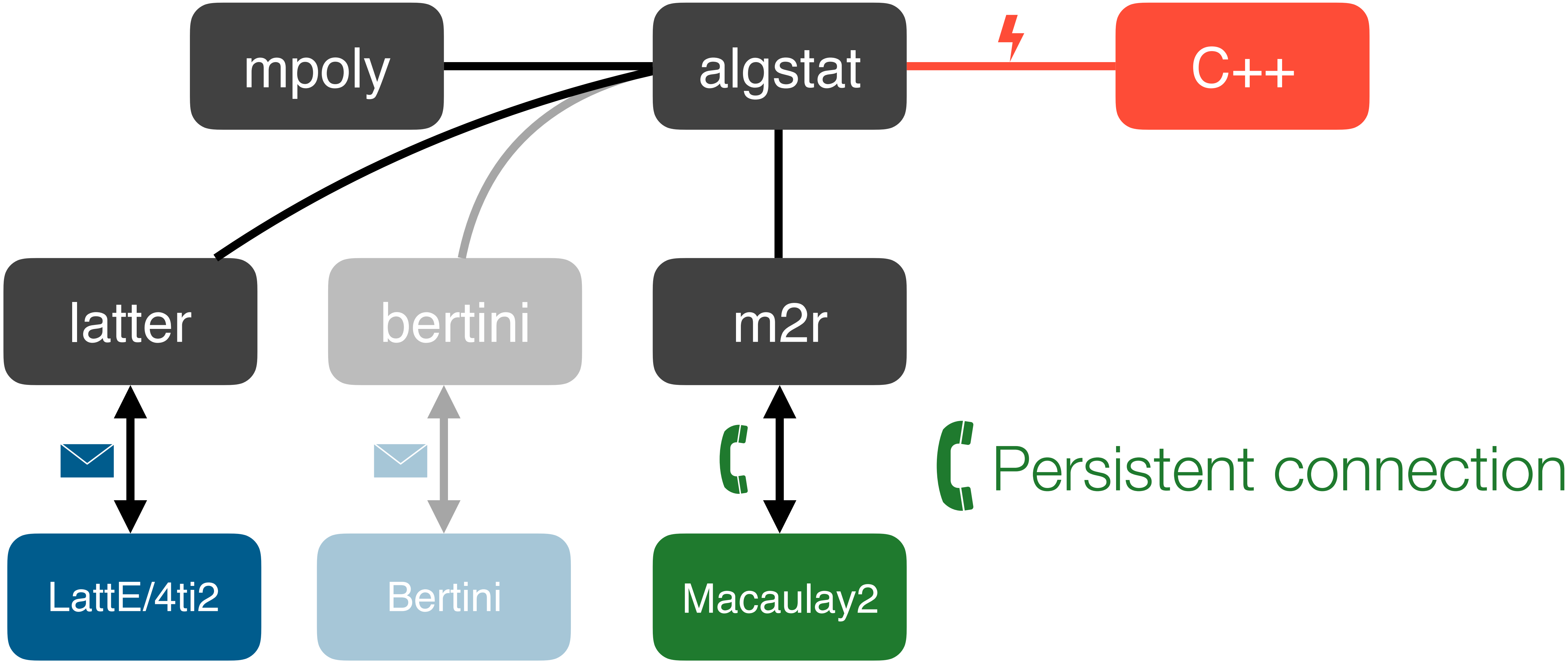
2014 : **algstat** – algebraic statistical data analysis



Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

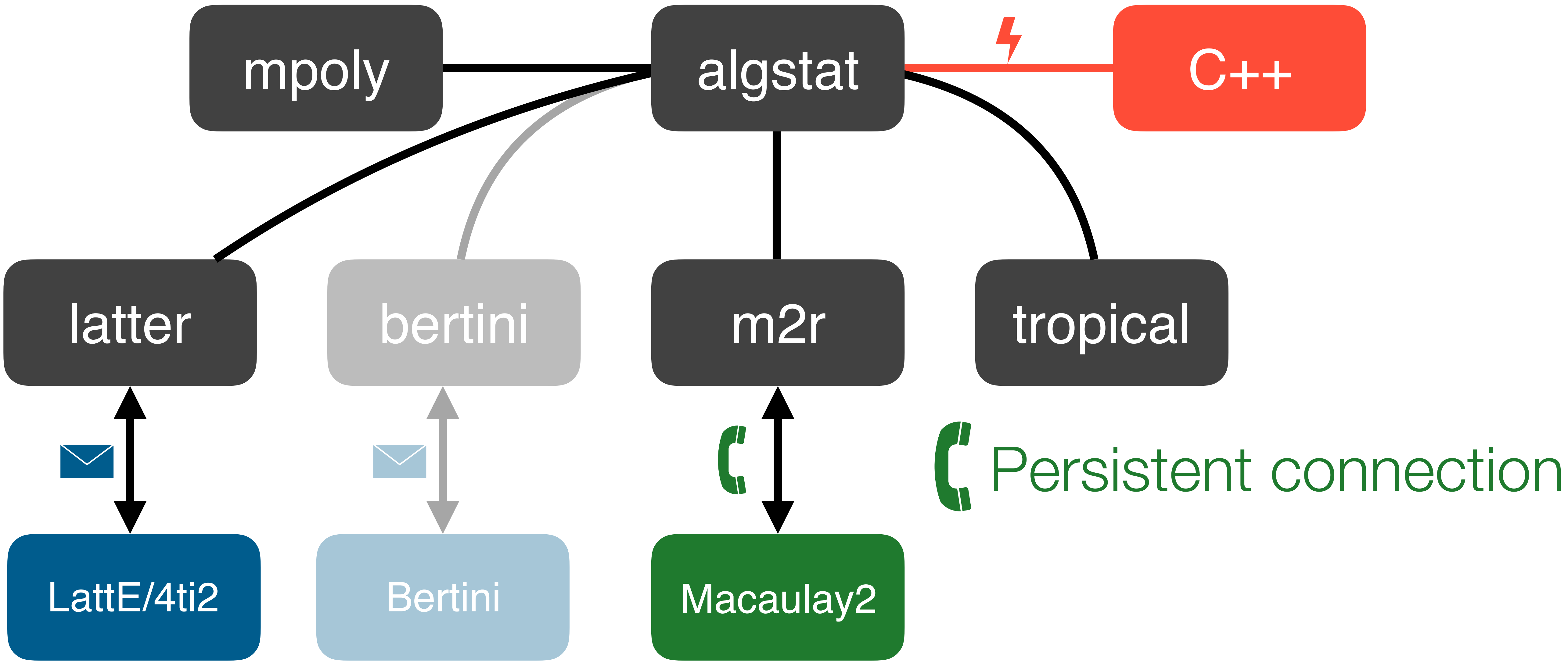
2014 : **algstat** – algebraic statistical data analysis



Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

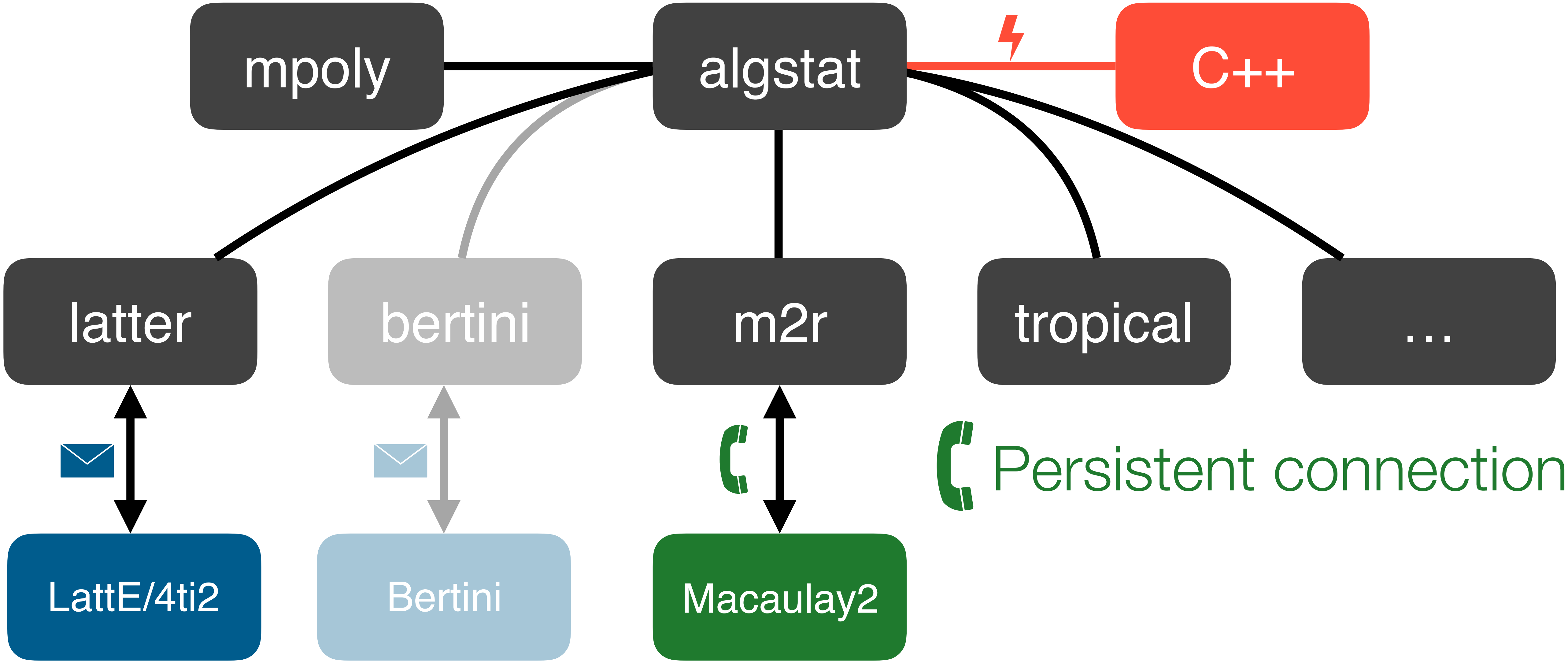
2014 : **algstat** – algebraic statistical data analysis



Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

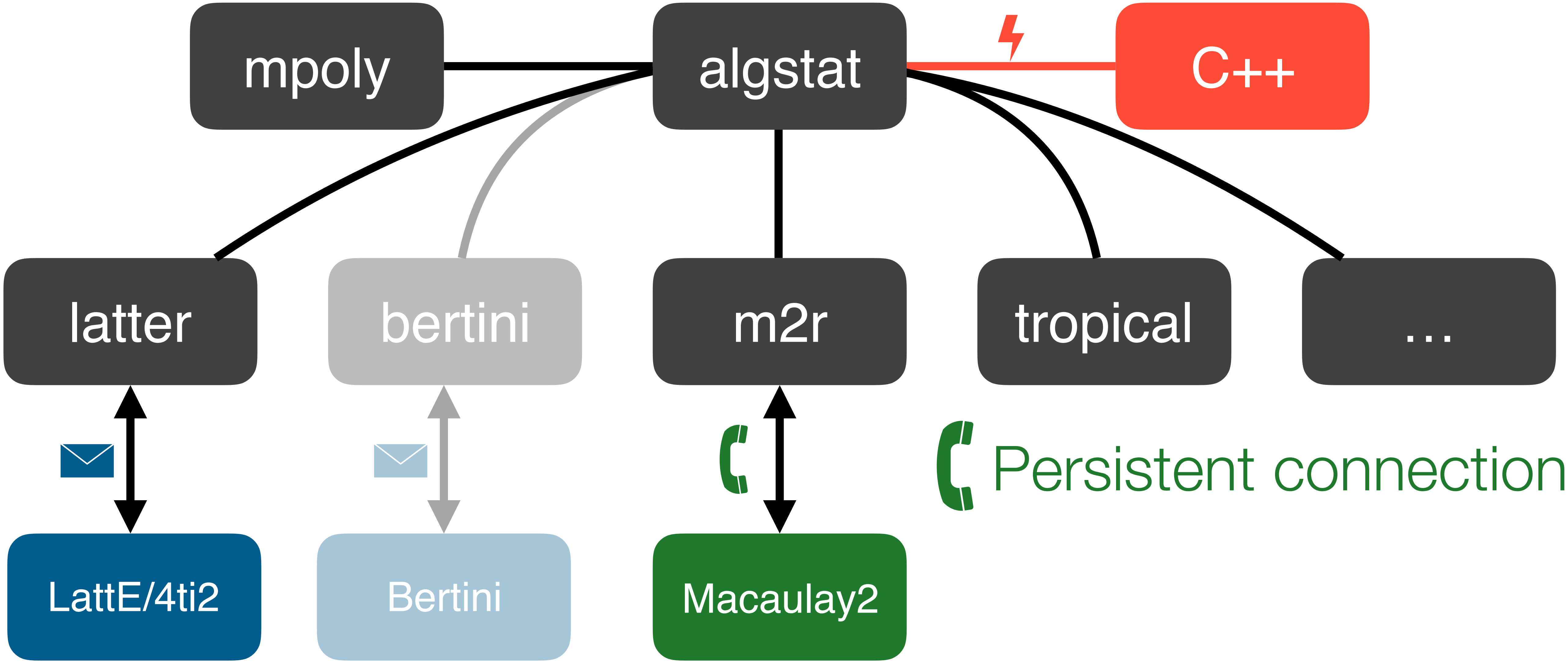
2014 : **algstat** – algebraic statistical data analysis



Historical perspective

2011 : **mpoly** – data structures and methods for multivariate polynomials

2014 : **algstat** – algebraic statistical data analysis



* Many other interconnections not shown

The algstat Ecosystem

Installing the **algstat** ecosystem

To install the packages I'm using in this talk, copy/paste the following code into R

```r
if(!requireNamespace("devtools"))
  install.packages("devtools")
devtools::install_github("dkahle/mpoly")
devtools::install_github("dkahle/latter")
devtools::install_github("dkahle/tropical")
devtools::install_github("coneill-math/m2r")
devtools::install_github("dkahle/algstat")
```

For **latter**, you'll need LattE/4ti2, which you can get here

For **algstat**, you'll want Bertini, which you can get here

mpoly

mpoly contains data structures and methods for polynomials

mpoly

mpoly contains data structures and methods for polynomials

Understanding **mpoly** requires understanding a bit about the R language

R’s main data structures

	homogeneous must hold the same kinds of objects	heterogeneous can hold different kinds of objects
1d	atomic vector int, num, char, logi, …	list
2d	matrix	data frame list of equal-length atomic vectors
nd	array	

R’s main data structures

R's main data structures

```r
(x <- c(1, 2, 3)) # a vector
# [1] 1 2 3
```

R’s main data structures

```
(x <- c(1, 2, 3)) # a vector
# [1] 1 2 3


(y <- list(1:3, c("a","b"), TRUE)) # a list
# [[1]]
# [1] 1 2 3
#
# [[2]]
# [1] "a" "b"
#
# [[3]]
# [1] TRUE
#
```

Names and classes

R objects can also have a list of metadata attached to them called attributes

Names and classes

R objects can also have a list of metadata attached to them called attributes

Two basic types of attributes are names and class

Names and classes

R objects can also have a list of metadata attached to them called attributes

Two basic types of attributes are names and class

- Names are used for easy, non-index based referencing

Names and classes

R objects can also have a list of metadata attached to them called attributes

Two basic types of attributes are names and class

- Names are used for easy, non-index based referencing
- Classes are R's most basic object oriented framework

Names and classes

R objects can also have a list of metadata attached to them called attributes

Two basic types of attributes are names and class

- Names are used for easy, non-index based referencing
- Classes are R's most basic object oriented framework

Classes are also used for method dispatch

(Methods are not contained in objects in R)

Names and classes

```
attr(x, "names") <- c("x", "y", "z")
x
# x y z
# 1 2 3
```

Names and classes

```
attr(x, "names") <- c("x", "y", "z")
x
# x y z
# 1 2 3

attr(x, "class") <- "foo"
x
# x y z
# 1 2 3
# attr(,"class")
# [1] "foo"
```

Names and classes

```
attr(x, "names") <- c("x", "y", "z")
x
# x y z
# 1 2 3

attr(x, "class") <- "foo"
x
# x y z
# 1 2 3
# attr(,"class")
# [1] "foo"

print(x)
# x y z
# 1 2 3
# attr(,"class")
# [1] "foo"
```

Names and classes

```
attr(x, "names") <- c("x", "y", "z")
x
# x y z
# 1 2 3

attr(x, "class") <- "foo"
x
# x y z
# 1 2 3
# attr(,"class")
# [1] "foo"

print(x)
# x y z
# 1 2 3
# attr(,"class")
# [1] "foo"
```

Typing `x` at the command line implicitly calls `print(x)`

Methods (S3 methods)

```
print(x)
# x y z
# 1 2 3
# attr(,"class")
# [1] "foo"

print.foo <- function(.) {
  cat( paste0(names(.), "^", .) )
}
x
# x^1 y^2 z^3
```

Methods (S3 methods)

```
print(x)
# x y z
# 1 2 3
# attr(,"class")
# [1] "foo"

print.foo <- function(.) {
  cat( paste0(names(.), "^", .) )
}
x
# x^1 y^2 z^3

str(x)
# Class 'foo'  Named num [1:3] 1 2 3
#   ..- attr(*, "names")= chr [1:3] "x" "y" "z"
```

mpoly's

mpoly's main basic data structure is the mpoly, a list of numeric vectors representing terms

mpoly's

mpoly's main basic data structure is the mpoly, a list of numeric vectors representing terms

```
library(mpoly)
```

mpoly's

mpoly's main basic data structure is the mpoly, a list of numeric vectors representing terms

```r
library(mpoly)
mp("x^2 + 2 x y - 1")
# x^2  +  2 x y  -  1
```

mpoly's

mpoly's main basic data structure is the mpoly,
a list of numeric vectors representing terms

```
library(mpoly)
mp("x^2 + 2 x y - 1")
# x^2  +  2 x y  -  1

str( mp("x^2 + 2 x y - 1") )
# List of 3
#  $ : Named num [1:2] 2 1
#   ..- attr(*, "names")= chr [1:2] "x" "coef"
#  $ : Named num [1:3] 1 1 2
#   ..- attr(*, "names")= chr [1:3] "x" "y" "coef"
#  $ : Named num -1
#   ..- attr(*, "names")= chr "coef"
#  - attr(*, "class")= chr "mpoly"
```

Methods

mpoly provides all basic arithmetic operations as well as well as related tools

Methods

mpoly provides all basic arithmetic operations as well as well as related tools

```
p <- mp("x + y"); q <- mp("x - y")
```

Methods

mpoly provides all basic arithmetic operations as well as well as related tools

```
p <- mp("x + y"); q <- mp("x - y")

p + q
# 2 x
```

Methods

mpoly provides all basic arithmetic operations as well as well as related tools

```
p <- mp("x + y"); q <- mp("x - y")

p + q
# 2 x
p * q
# x^2  -  y^2
```

Methods

mpoly provides all basic arithmetic operations as well as well as related tools

```
p <- mp("x + y"); q <- mp("x - y")

p + q
# 2 x
p * q
# x^2  -  y^2
p^2
# x^2  +  2 x y  +  y^2
```

Methods

mpoly provides all basic arithmetic operations as well as well as related tools

```
p <- mp("x + y"); q <- mp("x - y")

p + q
# 2 x
p * q
# x^2  -  y^2
p^2
# x^2  +  2 x y  +  y^2

f <- as.function(p, vector = FALSE)
# f(x, y)
f(1, 2)
# [1] 3
```

mpolyList’s

Vectors of mpolys are also defined with similar methods

mpolyList's

Vectors of mpolys are also defined with similar methods

```
(ps <- mp(c("x + y", "x - y^2")))
# x  +  y
# x  -  y^2
```

mpolyList's

Vectors of mpolys are also defined with similar methods

```
(ps <- mp(c("x + y", "x - y^2")))
# x  +  y
# x  -  y^2

g <- as.function(ps, vector = FALSE)
g(1, 2)
# [1]  3 -3
```

mpolyList's

Vectors of mpolys are also defined with similar methods

```
(ps <- mp(c("x + y", "x - y^2")))
# x  +  y
# x  -  y^2

g <- as.function(ps, vector = FALSE)
g(1, 2)
# [1]  3 -3
```

These kinds of lists are used by most other packages in the **algstat** ecosystem

latter

latter implements back-end connections to LattE/4ti2

latter

latter implements back-end connections to LattE/4ti2

It is used mostly for computing Markov and related bases

latter

```
library(latter)
#   LattE found in /Applications/latte/dest/bin
#   4ti2 found in /Applications/latte/dest/bin
```

latter

```
library(latter)
#  LattE found in /Applications/latte/dest/bin
#  4ti2 found in /Applications/latte/dest/bin

(A <- genmodel(c(2, 2), 1:2)) # 2x2 independence model
#      [,1] [,2] [,3] [,4]
# [1,]    1    0    1    0
# [2,]    0    1    0    1
# [3,]    1    1    0    0
# [4,]    0    0    1    1
```

latter

```
library(latter)
#  LattE found in /Applications/latte/dest/bin
#  4ti2 found in /Applications/latte/dest/bin

(A <- genmodel(c(2, 2), 1:2)) # 2x2 independence model
#      [,1] [,2] [,3] [,4]
# [1,]    1    0    1    0
# [2,]    0    1    0    1
# [3,]    1    1    0    0
# [4,]    0    0    1    1

markov(A)
#      [,1]
# [1,]    1
# [2,]   -1
# [3,]   -1
# [4,]    1
```

latter

```
library(latter)
#  LattE found in /Applications/latte/dest/bin
#  4ti2 found in /Applications/latte/dest/bin

(A <- genmodel(c(2, 2), 1:2)) # 2x2 independence model
#      [,1] [,2] [,3] [,4]
# [1,]    1    0    1    0
# [2,]    0    1    0    1
# [3,]    1    1    0    0
# [4,]    0    0    1    1
```

```
markov(A)
#      [,1]
# [1,]    1
# [2,]   -1
# [3,]   -1
# [4,]    1
```

```
graver(A)
#      [,1]
# [1,]    1
# [2,]   -1
# [3,]   -1
# [4,]    1
```

latter

latter

LattE functions are also available

latter

LattE functions are also available

```r
spec <- c("x + y <= 10", "x >= 1", "y >= 1")
```

latter

LattE functions are also available

```
spec <- c("x + y <= 10", "x >= 1", "y >= 1")
count(spec)
# [1] 45
```

latter

LattE functions are also available

```
spec <- c("x + y <= 10", "x >= 1", "y >= 1")
count(spec)
# [1] 45
count(spec, dilation = 10)
# [1] 3321
```

latter

LattE functions are also available

```
spec <- c("x + y <= 10", "x >= 1", "y >= 1")
count(spec)
# [1] 45
count(spec, dilation = 10)
# [1] 3321

latte_max(
  "-2 x + 3 y",
  c("x + y <= 10", "x >= 0", "y >= 0")
)
```

latter

LattE functions are also available

```
spec <- c("x + y <= 10", "x >= 1", "y >= 1")
count(spec)
# [1] 45
count(spec, dilation = 10)
# [1] 3321
latte_max(
  "-2 x + 3 y",
  c("x + y <= 10", "x >= 0", "y >= 0")
)
# $par
#  x  y
#  0 10
#
# $value
# [1] 30
```

m2r

m2r connections, data structures, and methods for Macaulay2

m2r

m2r connections, data structures, and methods for Macaulay2

Since Macaulay2 is a full computer algebra system, a persistent connection is needed

m2r

m2r connections, data structures, and methods for Macaulay2

Since Macaulay2 is a full computer algebra system, a persistent connection is needed

So… stay tuned for Chris O’Neill’s talk next!

bertini

Bertini was one of the original algstat connections, but it hasn’t been implemented in it’s “Version 2” form yet

bertini

Bertini was one of the original algstat connections, but it hasn’t been implemented in it’s “Version 2” form yet

```
library(algstat)
```

bertini

Bertini was one of the original algstat connections, but it hasn’t been implemented in it’s “Version 2” form yet

```
library(algstat)
polySolve(
  c("y == x^2", "y == 2 - x^2"),
  varOrder = c("x", "y")
)
# 2 solutions (x,y) found.
# (2 real, 0 complex; 2 nonsingular, 0 singular.)
#     (-1,1) (R)
#     ( 1,1) (R)
```

tropical

A new direction of **algstat** applications is phylogenetics

tropical

A new direction of **algstat** applications is phylogenetics

tropical is a (very!) new package intended to supply the necessary tropical geometry computations

tropical

A new direction of **algstat** applications is phylogenetics

tropical is a (very!) new package intended to supply the necessary tropical geometry computations

```
library(tropical)
# Using min-plus algebra.
```

tropical

A new direction of **algstat** applications is phylogenetics

tropical is a (very!) new package intended to supply the necessary tropical geometry computations

```
library(tropical)
# Using min-plus algebra.
```

$$x \oplus y = \min(x, y)$$

$$x \otimes y = x + y$$

tropical

A new direction of **algstat** applications is phylogenetics

tropical is a (very!) new package intended to supply the necessary tropical geometry computations

```
library(tropical)
# Using min-plus algebra.
```

$$x \oplus y = \min(x, y)$$

$$x \otimes y = x + y$$

Max-plus is available with `set_plus_max()`

tropical

```
# basic tropical arithmetic
```

tropical

```
# basic tropical arithmetic
1 %+% 5
# [1] 1
```

tropical

```
# basic tropical arithmetic
1 %+% 5
# [1] 1
1 %.% 5
# [1] 6
```

tropical

```
# basic tropical arithmetic
1 %+% 5
# [1] 1
1 %.% 5
# [1] 6
5 %^% 3
# [1] 15
```

tropical

```
# basic tropical arithmetic
1 %+% 5
# [1] 1
1 %.% 5
# [1] 6
5 %^% 3
# [1] 15


# vectorized for R-users
1:3 %+% 3:1
# [1] 1 2 1
1:3 %.% 3:1
# [1] 4 4 4
```

tropical

```
# basic tropical arithmetic
1 %+% 5
# [1] 1
1 %.% 5
# [1] 6
5 %^% 3
# [1] 15


# vectorized for R-users
1:3 %+% 3:1
# [1] 1 2 1
1:3 %.% 3:1
# [1] 4 4 4
```

```
# tropical mat. mult
```

tropical

```
# basic tropical arithmetic
1 %+% 5
# [1] 1
1 %.% 5
# [1] 6
5 %^% 3
# [1] 15


# vectorized for R-users
1:3 %+% 3:1
# [1] 1 2 1
1:3 %.% 3:1
# [1] 4 4 4
```

```
# tropical mat. mult
1:3 %..% 4:6
# [1] 5
```

tropical

```r
# basic tropical arithmetic
1 %+% 5
# [1] 1
1 %.% 5
# [1] 6
5 %^% 3
# [1] 15


# vectorized for R-users
1:3 %+% 3:1
# [1] 1 2 1
1:3 %.% 3:1
# [1] 4 4 4
```

```r
# tropical mat. mult
1:3 %..% 4:6
# [1] 5

(m1 <- matrix(1:6, 2, 3))
#      [,1] [,2] [,3]
# [1,]    1    3    5
# [2,]    2    4    6
(m2 <- matrix(6:1, 3, 2))
#      [,1] [,2]
# [1,]    6    3
# [2,]    5    2
# [3,]    4    1
```

tropical

```
# basic tropical arithmetic
1 %+% 5
# [1] 1
1 %.% 5
# [1] 6
5 %^% 3
# [1] 15


# vectorized for R-users
1:3 %+% 3:1
# [1] 1 2 1
1:3 %.% 3:1
# [1] 4 4 4
```

```
# tropical mat. mult
1:3 %..% 4:6
# [1] 5

(m1 <- matrix(1:6, 2, 3))
#      [,1] [,2] [,3]
# [1,]    1    3    5
# [2,]    2    4    6
(m2 <- matrix(6:1, 3, 2))
#      [,1] [,2]
# [1,]    6    3
# [2,]    5    2
# [3,]    4    1
m1 %..% m2
#      [,1] [,2]
# [1,]    7    4
# [2,]    8    5
```

algstat

```
data(politics)
politics
#             Party
# Personality Democrat Republican
#   Introvert        3          7
#   Extrovert        6          4
```

algstat

```
data(politics)
politics
#             Party
# Personality Democrat Republican
#   Introvert        3          7
#   Extrovert        6          4

(A <- hmat(c(2, 2), list(1, 2))) # alternative to genmodel
#     11 12 21 22
# 1+   1  1  0  0
# 2+   0  0  1  1
# +1   1  0  1  0
# +2   0  1  0  1
```

algstat

```
data(politics)
politics
#            Party
# Personality Democrat Republican
#   Introvert        3          7
#   Extrovert        6          4

(A <- hmat(c(2, 2), list(1, 2))) # alternative to genmodel
#     11 12 21 22
# 1+  1  1  0  0
# 2+  0  0  1  1
# +1  1  0  1  0
# +2  0  1  0  1

countTables(politics, A)
# [1] 10
```

algstat

```r
loglinear(~ Personality + Party, data = politics)
```

algstat

```
loglinear(~ Personality + Party, data = politics)
# Computing Markov moves (4ti2)... done.
# Running chain (C++)... done.
```

algstat

```
loglinear(~ Personality + Party, data = politics)
# Computing Markov moves (4ti2)... done.
# Running chain (C++)... done.
# Call:
# loglinear(model = ~Personality + Party, data = politics)
#
# Fitting method:
# Iterative proportional fitting (with stats::loglin)
#
# MCMC details:
# N = 10000 samples (after thinning), burn in = 1000, thinning = 10
#
#       Distance   Stat     SE p.value     SE mid.p.value
#        P(samp)                0.3677 0.0048      0.2201
#    Pearson X^2 1.8182 0.0146  0.3677 0.0048      0.2201
# Likelihood G^2 1.848  0.0155  0.3677 0.0048      0.2201
#  Freeman-Tukey 1.8749 0.0167  0.3677 0.0048      0.2201
#   Cressie-Read 1.8247 0.0148  0.3677 0.0048      0.2201
```

GitHub and Contributing

Contributions are welcome!

To submit a feature request or report a bug:

- Go to https://github.com/ and create a free account
- Go to https://github.com/dkahle/algstat
- Click *Issues*
- Click *New Issue*

Contributions are welcome!

To join the fray, submit a pull request (PR)!

- Go to https://github.com/ and create a free account
- Go to https://github.com/dkahle/algstat
- Click *Fork* to make your own copy of the repository
- In RStudio…
  - File > New Project… > Version Control > Git
  - Enter the URL of the repo, https://github.com/dkahle/algstat.git
  - Make changes to the code and commit them, see tutorial here
  - Push changes to GitHub
- On GitHub, click *Submit a Pull Request*

Upcoming Projects

Upcoming projects

tropical with Grant Innerst, Rudy Yoshida, Leon Zhang and Xu Zhang

m2r with Chris O'Neill and Jeff Sommars

algstat with Luis Garcia, Rudy Yoshida, …

bertini with Grant Innerst

…?

Thank you!!

www.kahle.io

https://github.com/dkahle/2017-SIAM-Talk

This material is based upon work supported by the National Science Foundation under Grant Nos. 1321794, 1622449, and 1622369.