Using Macaulay2 from within R: the m2r package

Christopher O'Neill

University of California Davis

coneill@math.ucdavis.edu

Joint with David Kahle and Jeff Sommars
Mathematics Research Communities on Algebraic Statistics

August 3, 2017

R and Macaulay2

R: a statistician's best friend

Data storage and manipulation, array calculations, data analysis, . . .

R and Macaulay2

R: a statistician's best friend

Data storage and manipulation, array calculations, data analysis, . . .

Macaulay2: an algebraic geometer's best friend

Polynomial ideals, Gröbner bases, Hilbert functions, . . .

R and Macaulay2

R: a statistician's best friend

Data storage and manipulation, array calculations, data analysis, . . .

Macaulay2: an algebraic geometer's best friend

Polynomial ideals, Gröbner bases, Hilbert functions, . . .

Algebraic statisticians: best of both worlds

Running Macaulay2 from R the old way

```
R version 3.3.0

...

>
```

Running Macaulay2 from R the old way

```
R version 3.3.0
...
> library("algstat")
>
```

Running Macaulay2 from R the old way

```
R version 3.3.0

...

> library("algstat")

> code <- "R = QQ[x,y,z]
           I = ideal(x^2, x*y, x^3*y^2)
           gens gb I"

>
```

Running Macaulay2 from R the old way

```
R version 3.3.0

...

> library("algstat")

> code <- "R = QQ[x,y,z]
           I = ideal(x^2, x*y, x^3*y^2)
           gens gb I"

> m2(code)
[1] "R"
[1] "ideal(x^2,x*y,x^3*y^2)"
[1] "matrix {{x*y, x^2}}"
```

Running Macaulay2 from R the old way

```
R version 3.3.0
...

> library("algstat")

> code <- "R = QQ[x,y,z]
           I = ideal(x^2, x*y, x^3*y^2)
           gens gb I"

> m2(code)
[1] "R"
[1] "ideal(x^2,x*y,x^3*y^2)"
[1] "matrix {{x*y, x^2}}"
```

m2Code.m2

```
f = "m2Out" << ""
f << toString( R = QQ[x,y,z] ) << endl
f << toString( I = ideal(x^2, x*y, x^3*y^2) ) << endl
f << toString( gens gb I ) << endl
f << close
```

The m2r package in action

```
R version 3.3.0
...
>
```

The m2r package in action

```
R version 3.3.0

...

> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 found in /usr/local/macaulay2/bin

>
```

The m2r package in action

```
R version 3.3.0
...

> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 found in /usr/local/macaulay2/bin

> start_m2()
Starting M2...  done.

>
```

The m2r package in action

```
R version 3.3.0
...

> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 found in /usr/local/macaulay2/bin

> start_m2()
Starting M2...  done.

> m2("1+1")
[1] "2"

>
```

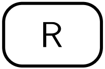
The m2r package in action

```
R version 3.3.0

...

> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 found in /usr/local/macaulay2/bin

> start_m2()
Starting M2...  done.

> m2("1+1")
[1] "2"

> m2("a = 5")
[1] "5"

>
```

The m2r package in action

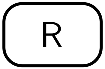
```
R version 3.3.0
...

> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 found in /usr/local/macaulay2/bin

> start_m2()
Starting M2...  done.

> m2("1+1")
[1] "2"

> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"
```

Under the hood: sockets



R

Under the hood: sockets



R

- `m2_start()`

Under the hood: sockets

R

M2

- `m2_start()`
- launch `M2` process

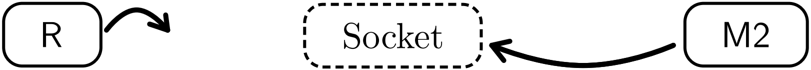
Under the hood: sockets

R

M2

- `m2_start()`
- launch `M2` process
- wait for available connection

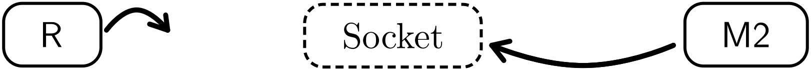
Under the hood: sockets



- `m2_start()`
- launch `M2` process
- wait for available connection

- create server socket

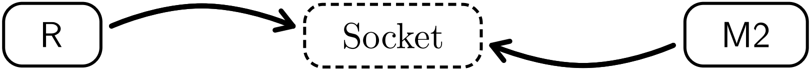
Under the hood: sockets



- `m2_start()`
- launch `M2` process
- wait for available connection

- create server socket
- wait for client connection

Under the hood: sockets

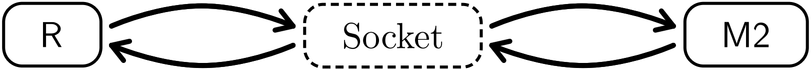


- `m2_start()`
- launch `M2` process
- wait for available connection

- create server socket
- wait for client connection

- connect to socket

Under the hood: sockets

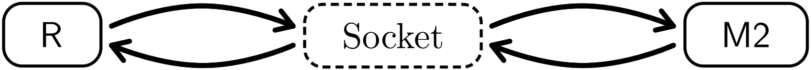


- `m2_start()`
- launch `M2` process
- wait for available connection

- create server socket
- wait for client connection

- connect to socket

Under the hood: sockets

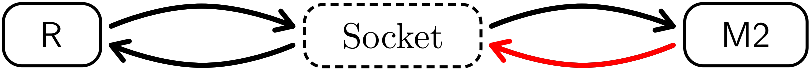


- `m2_start()`
- launch `M2` process
- wait for available connection

- create server socket
- wait for client connection

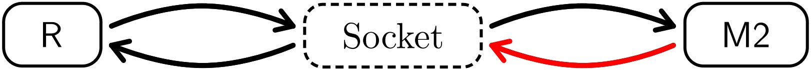
- connect to socket
- wait for message from server

Under the hood: sockets



- `m2_start()`
- launch `M2` process
- wait for available connection
- create server socket
- wait for client connection
- connect to socket
- wait for message from server
- send `"1.0.0"`

Under the hood: sockets



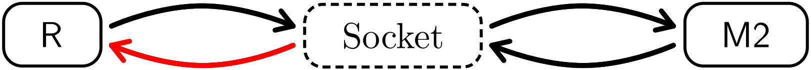
- `m2_start()`
- launch `M2` process
- wait for available connection

- create server socket
- wait for client connection

- connect to socket
- wait for message from server

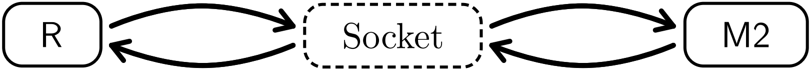
- send `"1.0.0"`
- wait for input from client

Under the hood: sockets



- `m2_start()`
- launch `M2` process
- wait for available connection
- create server socket
- wait for client connection
- connect to socket
- wait for message from server
- send `"1.0.0"`
- wait for input from client
- receive `"1.0.0"`

Under the hood: sockets



- `m2_start()`
- launch `M2` process
- wait for available connection

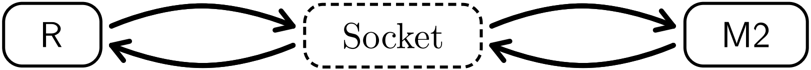
- create server socket
- wait for client connection

- connect to socket
- wait for message from server

- send `"1.0.0"`
- wait for input from client

- receive `"1.0.0"`
- verify version match

Under the hood: sockets



- `m2_start()`
- launch `M2` process
- wait for available connection

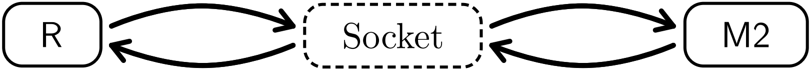
- create server socket
- wait for client connection

- connect to socket
- wait for message from server

- send `"1.0.0"`
- wait for input from client

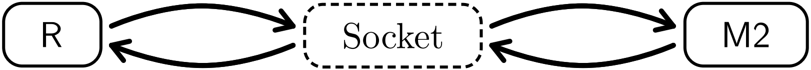
- receive `"1.0.0"`
- verify version match
- return from `m2_start()`

Under the hood: sockets



- wait for input from client

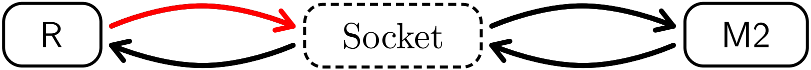
Under the hood: sockets



- wait for input from client

- `m2("1+1")`

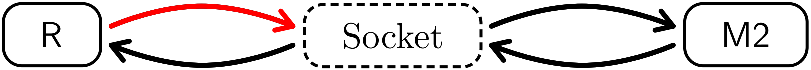
Under the hood: sockets



- wait for input from client

- `m2("1+1")`
- send `"1+1"` to server

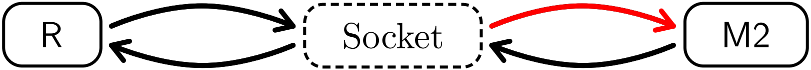
Under the hood: sockets



- wait for input from client

- `m2("1+1")`
- send `"1+1"` to server
- wait for response

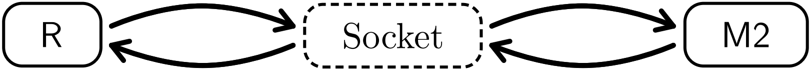
Under the hood: sockets



- wait for input from client
- receive "1+1" from client

- m2("1+1")
- send "1+1" to server
- wait for response

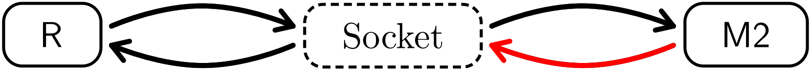
Under the hood: sockets



- `m2("1+1")`
- send `"1+1"` to server
- wait for response

- wait for input from client
- receive `"1+1"` from client
- evaluate `"1+1"` to `"2"`

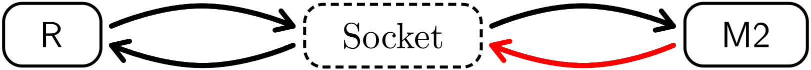
Under the hood: sockets



- `m2("1+1")`
- send `"1+1"` to server
- wait for response

- wait for input from client
- receive `"1+1"` from client
- evaluate `"1+1"` to `"2"`
- send `"2"` to client

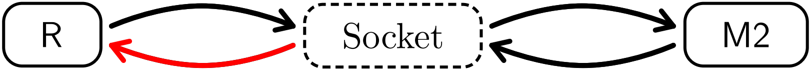
Under the hood: sockets



- m2("1+1")
- send "1+1" to server
- wait for response

- wait for input from client
- receive "1+1" from client
- evaluate "1+1" to "2"
- send "2" to client
- wait for input from client

Under the hood: sockets



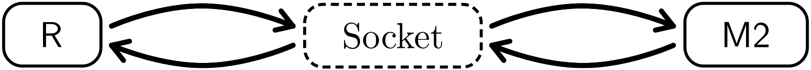
- wait for input from client

- `m2("1+1")`
- send `"1+1"` to server
- wait for response

- receive `"1+1"` from client
- evaluate `"1+1"` to `"2"`
- send `"2"` to client
- wait for input from client

- receive `"2"` from server

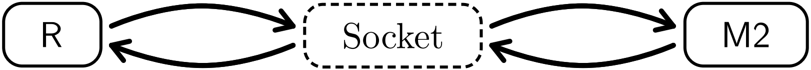
Under the hood: sockets



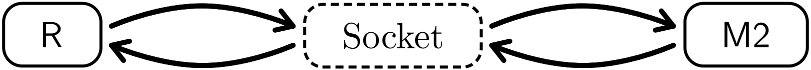
- m2("1+1")
- send "1+1" to server
- wait for response
- receive "2" from server
- return "2" from m2()

- wait for input from client
- receive "1+1" from client
- evaluate "1+1" to "2"
- send "2" to client
- wait for input from client

Under the hood: sockets

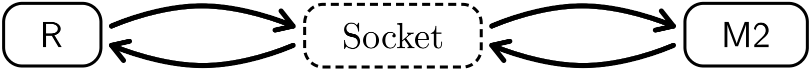


Under the hood: sockets



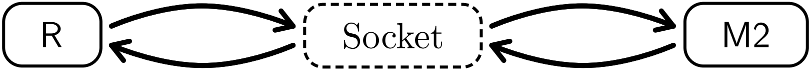
>

Under the hood: sockets



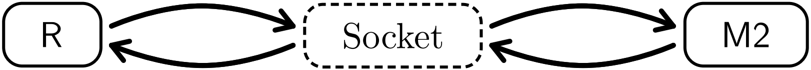
```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

>
```

Under the hood: sockets



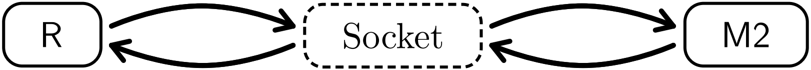
```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> m2("1+")
Error: Macaulay2 Error!

>
```

Under the hood: sockets



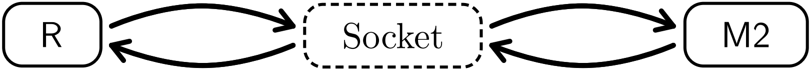
```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> m2("1+")
Error: Macaulay2 Error!

> m2("2+1")
[1] "3"
```

Under the hood: sockets



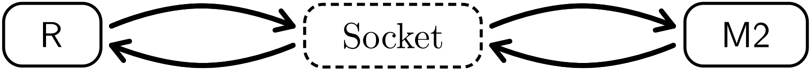
```
>
```

Under the hood: sockets



```
> start_m2()
Starting M2...  done.

> m2("1+1")
[1] "2"

>
```

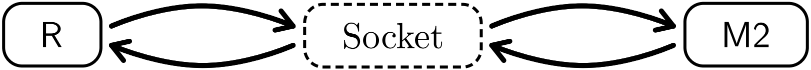
Under the hood: sockets



```
> start_m2()
Starting M2... done.

> m2("1+1")
[1] "2"

>
```

So. . . now what?

Under the hood: sockets



```
> start_m2()
Starting M2... done.

> m2("1+1")
[1] "2"

>
```

So... now what?

New features since creation:

- Lots of convenience functions
- High-level parser
- Cloud computing

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

>
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> m2("R = QQ[x,y,z]")
[1] "QQ(monoid[x..z, Degrees => {3:1}, Heft => {1}, MonomialOrder
=> VerticalList{MonomialSize => 32, GRevLex => {3:1}, Position =>
Up}, DegreeRank => 1])"

>
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> m2("R = QQ[x,y,z]")
[1] "QQ(monoid[x..z, Degrees => {3:1}, Heft => {1}, MonomialOrder
=> VerticalList{MonomialSize => 32, GRevLex => {3:1}, Position =>
Up}, DegreeRank => 1])"

> m2("I = ideal(x^2, x*y, x^3*y^2)")
[1] "ideal map((R)^1,(R)^{{-2},{-2},{-5}},{{x^2, x*y, x^3*y^2}})"

>
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> m2("R = QQ[x,y,z]")
[1] "QQ(monoid[x..z, Degrees => {3:1}, Heft => {1}, MonomialOrder
=> VerticalList{MonomialSize => 32, GRevLex => {3:1}, Position =>
Up}, DegreeRank => 1])"

> m2("I = ideal(x^2, x*y, x^3*y^2)")
[1] "ideal map((R)^1,(R)^{{-2},{-2},{-5}},{{x^2, x*y, x^3*y^2}})"

> m2("gens gb I")
[1] "map((R)^1,(R)^{{-2},{-2}},{{x*y, x^2}})"
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> m2("R = QQ[x,y,z]")
[1] "QQ(monoid[x..z, Degrees => {3:1}, Heft => {1}, MonomialOrder
=> VerticalList{MonomialSize => 32, GRevLex => {3:1}, Position =>
Up}, DegreeRank => 1])"

> m2("I = ideal(x^2, x*y, x^3*y^2)")
[1] "ideal map((R)^1,(R)^{{-2},{-2},{-5}},{{x^2, x*y, x^3*y^2}})"

> m2("gens gb I")
[1] "map((R)^1,(R)^{{-2},{-2}},{{x*y, x^2}})"
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

>
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> (R <- ring("x", "y", "z", coefring = "QQ"))
M2 Ring:  QQ[x,y,z], grevlex order

>
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> (R <- ring("x", "y", "z", coefring = "QQ"))
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

>
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> (R <- ring("x", "y", "z", coefring = "QQ"))
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> (mygens <- gb(I))
x y
x^2

>
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> (R <- ring("x", "y", "z", coefring = "QQ"))
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> (mygens <- gb(I))
x y
x^2

> mygens[[2]]
x^2
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> (R <- ring("x", "y", "z", coefring = "QQ"))
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> (mygens <- gb(I))
x y
x^2

> mygens[[2]]          ⟵ mpoly
x^2
```

Convenience functions

```
> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"

> (R <- ring("x", "y", "z", coefring = "QQ"))
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> (mygens <- gb(I))     ⟵ mpolylist
x y
x^2

> mygens[[2]]           ⟵ mpoly
x^2
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> radical(I)
M2 Ideal of ring QQ[x,y,z] (grevlex) with generator :
< x >

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> radical(I)
M2 Ideal of ring QQ[x,y,z] (grevlex) with generator :
< x >

> saturate(I,ideal("x^5"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generator :
< 1 >

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> radical(I)
M2 Ideal of ring QQ[x,y,z] (grevlex) with generator :
< x >

> saturate(I,ideal("x^5"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generator :
< 1 >

> I+I
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2, x^2, x y, x^3 y^2 >

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> radical(I)
M2 Ideal of ring QQ[x,y,z] (grevlex) with generator :
< x >

> saturate(I,ideal("x^5"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generator :
< 1 >

> I+I
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2, x^2, x y, x^3 y^2 >

> gb(I+I)
x y
x^2
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> primary_decomposition(I)
M2 List of ideals of QQ[x,y,z] (grevlex) :
< x >
< x^2, y >

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> primary_decomposition(I)
M2 List of ideals of QQ[x,y,z] (grevlex) :
< x >
< x^2, y >

> dimension(I)
[1] 2

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> primary_decomposition(I)
M2 List of ideals of QQ[x,y,z] (grevlex) :
< x >
< x^2, y >

> dimension(I)
[1] 2

> ring("x", "y", "z", coefring = "QQ", code = TRUE)
m2rintring00000002 = QQ[x,y,z,MonomialOrder=>{GRevLex=>3}]

>
```

Convenience functions

```
> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> primary_decomposition(I)
M2 List of ideals of QQ[x,y,z] (grevlex) :
< x >
< x^2, y >

> dimension(I)
[1] 2

> ring("x", "y", "z", coefring = "QQ", code = TRUE)
m2rintring00000002 = QQ[x,y,z,MonomialOrder=>{GRevLex=>3}]

> dimension(I, code = TRUE)
dim(m2rintideal00000001)
```

The parser

```
>
```

The parser

```
> m2_matrix(matrix(c(1,2,3,4), nrow = 2, ncol = 2))
     [,1] [,2]
[1,]    1    3
[2,]    2    4
M2 Matrix over ZZ[]

>
```

The parser

```
> m2_matrix(matrix(c(1,2,3,4), nrow = 2, ncol = 2))
     [,1] [,2]
[1,]    1    3
[2,]    2    4
M2 Matrix over ZZ[]

> m2_matrix(matrix(c(1,2,3,4), nrow = 2, ncol = 2), code = TRUE)
m2rintmatrix00000002 = matrix {{(1),(3)},{(2),(4)}}

>
```

The parser

```
> m2_matrix(matrix(c(1,2,3,4), nrow = 2, ncol = 2))
     [,1] [,2]
[1,]    1    3
[2,]    2    4
M2 Matrix over ZZ[]

> m2_matrix(matrix(c(1,2,3,4), nrow = 2, ncol = 2), code = TRUE)
m2rintmatrix00000002 = matrix {{(1),(3)},{(2),(4)}}

> m2("m2rintmatrix00000002 = matrix {{(1),(3)},{(2),(4)}}")
[1] "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"

>
```

The parser

```
> m2_matrix(matrix(c(1,2,3,4), nrow = 2, ncol = 2))
     [,1] [,2]
[1,]    1    3
[2,]    2    4
M2 Matrix over ZZ[]
```

```
> m2_matrix(matrix(c(1,2,3,4), nrow = 2, ncol = 2), code = TRUE)
m2rintmatrix00000002 = matrix {{(1),(3)},{(2),(4)}}
```

```
> m2("m2rintmatrix00000002 = matrix {{(1),(3)},{(2),(4)}}")
[1] "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"
```

```
> m2_parse("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
     [,1] [,2]
[1,]    1    3
[2,]    2    4
M2 Matrix over ZZ[]
```

The parser

Parsing "`map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})`"

The parser

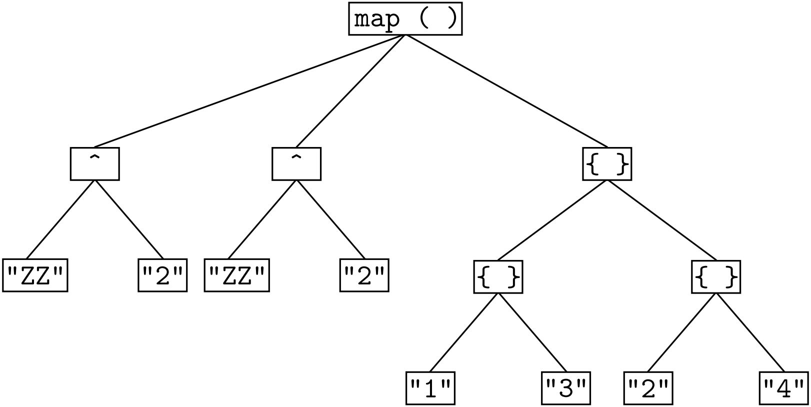
Parsing "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"

```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ" 
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"  
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"  
```

The parser

Parsing "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"

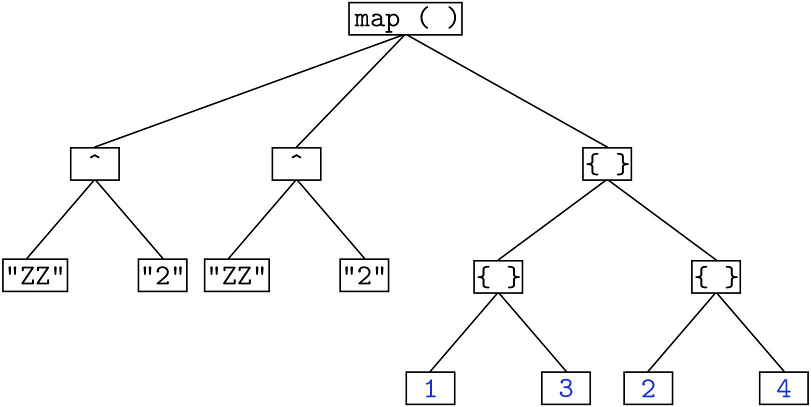
```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ"
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"
```



The parser

Parsing `"map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"`

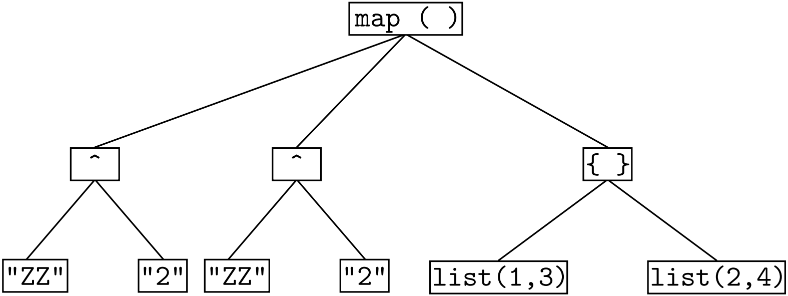
```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ"
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"
```



The parser

Parsing "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"

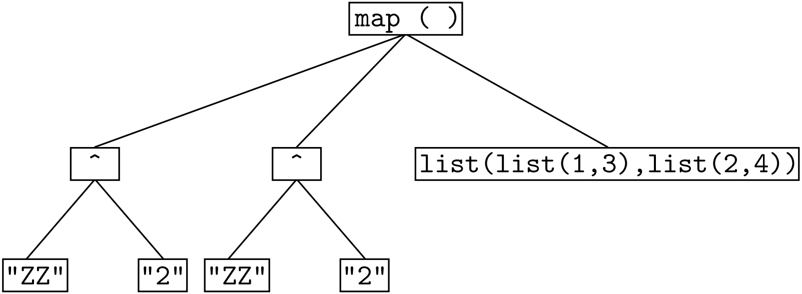
```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ"
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"
```



The parser

Parsing "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"

```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ"
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"
```



map ( )

^

^

list(list(1,3),list(2,4))

"ZZ"

"2"

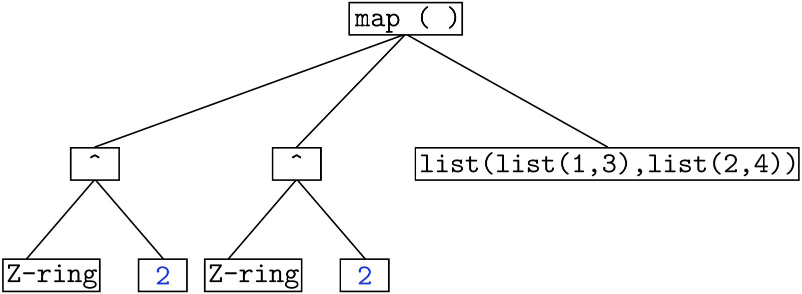
"ZZ"

"2"

The parser

Parsing "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"

```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ"
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"
```



map ( )

^

^

list(list(1,3),list(2,4))

Z-ring

2

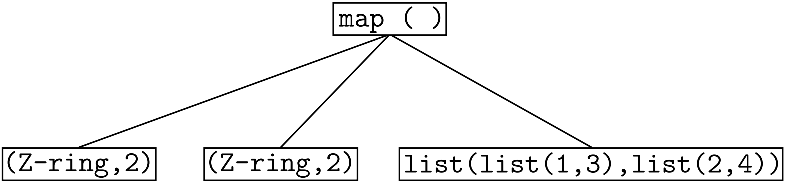
Z-ring

2

The parser

Parsing `"map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"`

```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ"
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"
```



map ( )

(Z-ring,2)

(Z-ring,2)

list(list(1,3),list(2,4))

The parser

Parsing "map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})"

```
> m2_tokenize("map((ZZ)^2,(ZZ)^2,{{1, 3}, {2, 4}})")
 [1] "map" "("   "("   "ZZ"  ")"   "^"   "2"   ","   "("   "ZZ"
[11] ")"   "^"   "2"   ","   "{"   "{"   "1"   ","   "3"   "}"
[21] ","   "{"   "2"   ","   "4"   "}"   "}"   ")"
```

```
               [,1] [,2]
          [1,]    1    3
          [2,]    2    4
M2 Matrix over ZZ[]
```

The parser

```
>
```

The parser

```
> m2_parse(m2("x"))
M2 Symbol: x

>
```

The parser

```
> m2_parse(m2("x"))
M2 Symbol: x

> m2_parse(m2("ZZ"))
M2 Ring: ZZ[], grevlex order

>
```

The parser

```
> m2_parse(m2("x"))
M2 Symbol: x

> m2_parse(m2("ZZ"))
M2 Ring: ZZ[], grevlex order

> m2("m2rintring00000002 = QQ[x,y,z,MonomialOrder=>{GRevLex=>3}]")
[1] "QQ(monoid[x..z, Degrees => {3:1}, Heft => {1}, MonomialOrder
=> VerticalList{MonomialSize => 32, GRevLex => {3:1}, Position =>
Up}, DegreeRank => 1])"

>
```

The parser

```
> m2_parse(m2("x"))
M2 Symbol: x

> m2_parse(m2("ZZ"))
M2 Ring: ZZ[], grevlex order

> m2("m2rintring00000002 = QQ[x,y,z,MonomialOrder=>{GRevLex=>3}]")
[1] "QQ(monoid[x..z, Degrees => {3:1}, Heft => {1}, MonomialOrder
=> VerticalList{MonomialSize => 32, GRevLex => {3:1}, Position =>
Up}, DegreeRank => 1])"

> m2_parse(m2("m2rintring00000002"))
M2 Ring: QQ[x,y,z], grevlex order

>
```

The parser

```
> m2_parse(m2("x"))
M2 Symbol: x

> m2_parse(m2("ZZ"))
M2 Ring: ZZ[], grevlex order

> m2("m2rintring00000002 = QQ[x,y,z,MonomialOrder=>{GRevLex=>3}]")
[1] "QQ(monoid[x..z, Degrees => {3:1}, Heft => {1}, MonomialOrder
=> VerticalList{MonomialSize => 32, GRevLex => {3:1}, Position =>
Up}, DegreeRank => 1])"

> m2_parse(m2("m2rintring00000002"))
M2 Ring: QQ[x,y,z], grevlex order

> m2("ideal({x^2+2*x,2*x+3})")
[1] "ideal map((m2rintring00000002)^1,(m2rintring00000002)^{{-2},
{-1}},{{x^2+2*x, 2*x+3}})"
```

Parser “extensibility”

```
> m2("ideal({x^2+2*x,2*x+3})")
[1] "ideal map((m2rintring00000002)^1,(m2rintring00000002)^{{-2},
{-1}},{{x^2+2*x, 2*x+3}})"
```

Parser “extensibility”

```
> m2("ideal({x^2+2*x,2*x+3})")
[1] "ideal map((m2rintring00000002)^1,(m2rintring00000002)^{{-2},
{-1}},{{x^2+2*x, 2*x+3}})"

m2_parse_function.m2_map <- function(x) {
  R1 <- x[[1]]
  R2 <- x[[2]]
      ⋮
  m2_structure(
    mat,
    m2_name = "",
    m2_class = "m2_matrix",
    m2_meta = list(
      ring = R1
    ),
    base_class = "matrix"
  )
}
```

Parser "extensibility"

```
> m2("ideal({x^2+2*x,2*x+3})")
[1] "ideal map((m2rintring00000002)^1,(m2rintring00000002)^{{-2},
{-1}},{{x^2+2*x, 2*x+3}})"
```

Parser “extensibility”

```
> m2("ideal({x^2+2*x,2*x+3})")
[1] "ideal map((m2rintring00000002)^1,(m2rintring00000002)^{{-2},
{-1}},{{x^2+2*x, 2*x+3}})"

m2_parse_function.m2_ideal <- function(x) {
  m2_structure(
    m2_name = "",
    m2_class = "m2_ideal",
    m2_meta = list(
      ring = m2_meta(x[[1]], "ring"),
      gens = structure(x[[1]][1,], class = "mpolyList")
    )
  )
}
```

m2r is now in the cloud!

```
>
```

m2r is now in the cloud!

```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

>
```

m2r is now in the cloud!

```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

> start_m2()
Connecting to M2 in the cloud...
done.

>
```

m2r is now in the cloud!

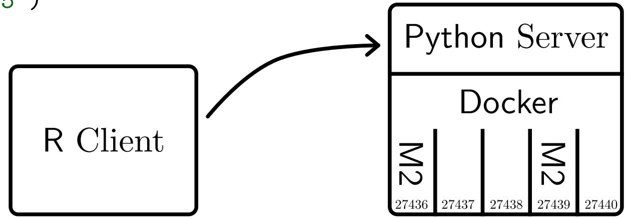
```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

> start_m2()
Connecting to M2 in the cloud...
done.

> m2("a = 5")
[1] "5"

>
```

m2r is now in the cloud!

```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

> start_m2()
Connecting to M2 in the cloud...
done.

> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"
```

m2r is now in the cloud!

```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

> start_m2()
Connecting to M2 in the cloud...
done.

> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"
```



R Client

Python Server

Docker

M2 27436

27437

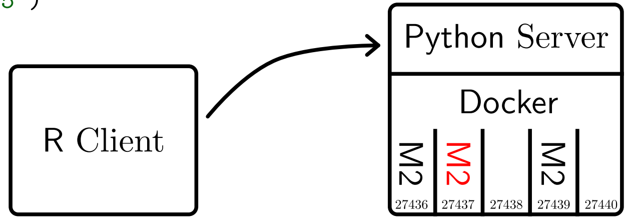
27438

M2 27439

27440

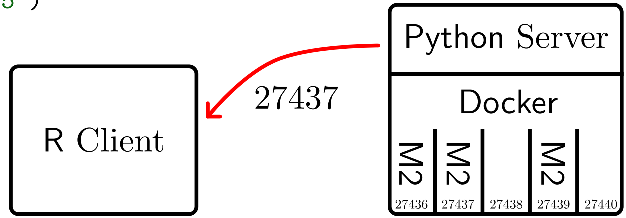
m2r is now in the cloud!

```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

> start_m2()
Connecting to M2 in the cloud...
done.

> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"
```



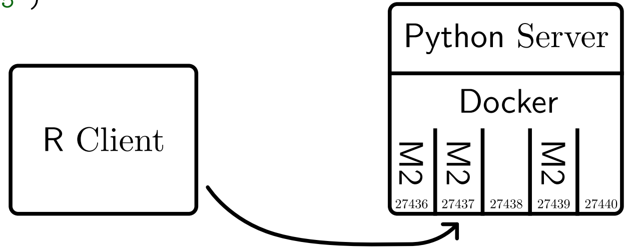
m2r is now in the cloud!

```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

> start_m2()
Connecting to M2 in the cloud...
done.

> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"
```



m2r is now in the cloud!

```
> library("m2r")
Loading required package: mpoly
Loading required package: stringr
please cite mpoly if you use it; see citation("mpoly")
M2 not found; defaulting to cloud.
Use set_m2r_path("/path/to/m2") to run M2 locally.

> start_m2()
Connecting to M2 in the cloud...
done.

> m2("a = 5")
[1] "5"

> m2("a")
[1] "5"
```



R Client

Python Server

Docker

M2 M2 M2

27436 27437 27438 27439 27440

More fancy features out there: reference functions

```
>
```

More fancy features out there: reference functions

```
> R <- ring("x", "y", "z", coefring = "QQ")
M2 Ring:  QQ[x,y,z], grevlex order

>
```

More fancy features out there: reference functions

```
> R <- ring("x", "y", "z", coefring = "QQ")
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

>
```

More fancy features out there: reference functions

```
> R <- ring("x", "y", "z", coefring = "QQ")
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> gb(I)
x y
x^2

>
```

More fancy features out there: reference functions

```
> R <- ring("x", "y", "z", coefring = "QQ")
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> gb(I)
x y
x^2

> (J <- ideal.("x^2", "x*y", "x^3*y^2"))
M2 Pointer Object
  ExternalString : ideal map((m2rintring00000001)^1,(m2rin...
         M2 Name : m2rintideal00000004
        M2 Class : Ideal (Type)

>
```

More fancy features out there: reference functions

```
> R <- ring("x", "y", "z", coefring = "QQ")
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> gb(I)
x y
x^2

> (J <- ideal.("x^2", "x*y", "x^3*y^2"))
M2 Pointer Object
  ExternalString : ideal map((m2rintring00000001)^1,(m2rin...
         M2 Name : m2rintideal00000004
        M2 Class : Ideal (Type)

> gb(J)
x y
x^2
```

More fancy features out there: reference functions

```
> R <- ring("x", "y", "z", coefring = "QQ")
M2 Ring:  QQ[x,y,z], grevlex order

> (I <- ideal("x^2", "x*y", "x^3*y^2"))
M2 Ideal of ring QQ[x,y,z] (grevlex) with generators :
< x^2, x y, x^3 y^2 >

> gb(I)
x y
x^2

> (J <- ideal.("x^2", "x*y", "x^3*y^2"))
M2 Pointer Object
  ExternalString : ideal map((m2rintring00000001)^1,(m2rin...
         M2 Name : m2rintideal00000004
        M2 Class : Ideal (Type)

> gb(J)
x y
x^2
```

m2_parse(J) ⟶ I

Thank you MRC!

References

D. Kahle, C. O'Neill, and J. Sommars (2017)
A computer algebra system for R: Macaulay2 and the m2r package
submitted. Available at [arXiv:1706.07797].

D. Grayson and M. Stillman (2006)
Macaulay2, a software system for research in algebraic geometry,
available at `http://www.math.uiuc.edu/Macaulay2/`.

D. Kahle (2013)
mpoly: Multivariate polynomials in R
The R Journal 5 (1), 162–170.

R Core Team (2014)
R: A language and environment for statistical computing
R Foundation for Statistical Computing, Vienna, Austria

References

D. Kahle, C. O'Neill, and J. Sommars (2017)
A computer algebra system for R: Macaulay2 and the m2r package
submitted. Available at [arXiv:1706.07797].

D. Grayson and M. Stillman (2006)
Macaulay2, a software system for research in algebraic geometry,
available at `http://www.math.uiuc.edu/Macaulay2/`.

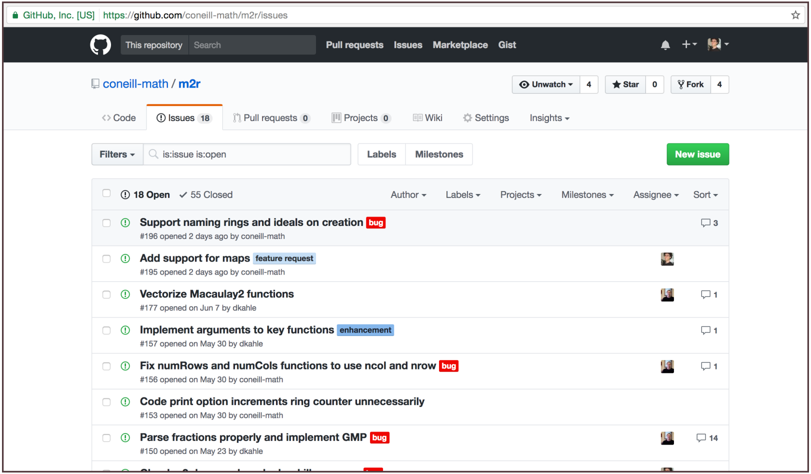
D. Kahle (2013)
mpoly: Multivariate polynomials in R
The R Journal 5 (1), 162–170.

R Core Team (2014)
R: A language and environment for statistical computing
R Foundation for Statistical Computing, Vienna, Austria

Thanks!

YOU should request lots of features!

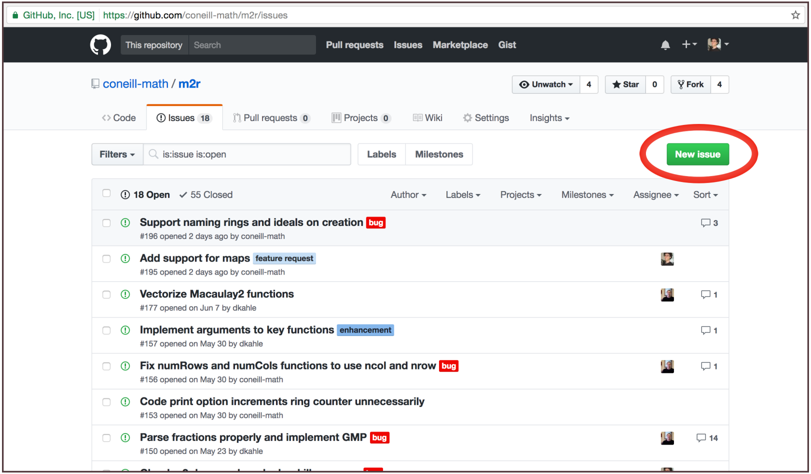
`https://github.com/coneill-math/m2r`



- □ Support naming rings and ideals on creation **bug** — #196 opened 2 days ago by coneill-math — 3
- □ Add support for maps **feature request** — #195 opened 2 days ago by coneill-math
- □ Vectorize Macaulay2 functions — #177 opened on Jun 7 by dkahle — 1
- □ Implement arguments to key functions **enhancement** — #157 opened on May 30 by dkahle — 1
- □ Fix numRows and numCols functions to use ncol and nrow **bug** — #156 opened on May 30 by coneill-math — 1
- □ Code print option increments ring counter unnecessarily — #153 opened on May 30 by coneill-math
- □ Parse fractions properly and implement GMP **bug** — #150 opened on May 23 by dkahle — 14

YOU should request lots of features!

`https://github.com/coneill-math/m2r`

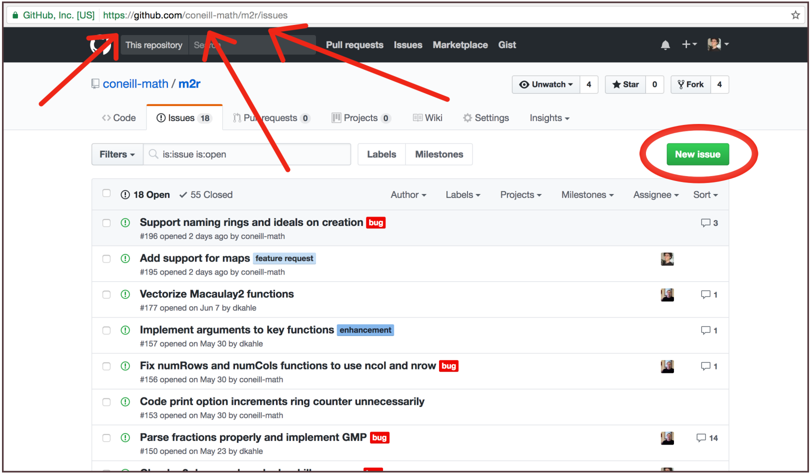


18 Open ✓ 55 Closed

- Support naming rings and ideals on creation `bug` — #196 opened 2 days ago by coneill-math — 3
- Add support for maps `feature request` — #195 opened 2 days ago by coneill-math
- Vectorize Macaulay2 functions — #177 opened on Jun 7 by dkahle — 1
- Implement arguments to key functions `enhancement` — #157 opened on May 30 by dkahle — 1
- Fix numRows and numCols functions to use ncol and nrow `bug` — #156 opened on May 30 by coneill-math — 1
- Code print option increments ring counter unnecessarily — #153 opened on May 30 by coneill-math
- Parse fractions properly and implement GMP `bug` — #150 opened on May 23 by dkahle — 14

YOU should request lots of features!

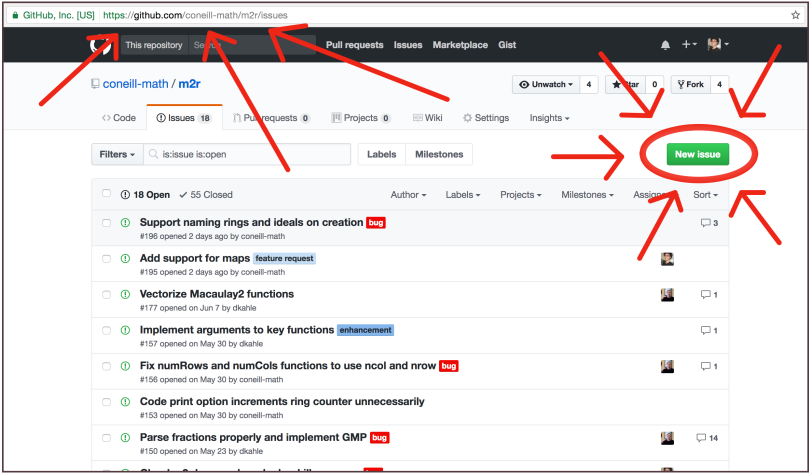
`https://github.com/coneill-math/m2r`



- Support naming rings and ideals on creation `bug` — 3
  #196 opened 2 days ago by coneill-math
- Add support for maps `feature request`
  #195 opened 2 days ago by coneill-math
- Vectorize Macaulay2 functions — 1
  #177 opened on Jun 7 by dkahle
- Implement arguments to key functions `enhancement` — 1
  #157 opened on May 30 by dkahle
- Fix numRows and numCols functions to use ncol and nrow `bug` — 1
  #156 opened on May 30 by coneill-math
- Code print option increments ring counter unnecessarily
  #153 opened on May 30 by coneill-math
- Parse fractions properly and implement GMP `bug` — 14
  #150 opened on May 23 by dkahle

YOU should request lots of features!

`https://github.com/coneill-math/m2r`



- □ Support naming rings and ideals on creation **bug** — #196 opened 2 days ago by coneill-math — 3
- □ Add support for maps **feature request** — #195 opened 2 days ago by coneill-math
- □ Vectorize Macaulay2 functions — #177 opened on Jun 7 by dkahle — 1
- □ Implement arguments to key functions **enhancement** — #157 opened on May 30 by dkahle — 1
- □ Fix numRows and numCols functions to use ncol and nrow **bug** — #156 opened on May 30 by coneill-math — 1
- □ Code print option increments ring counter unnecessarily — #153 opened on May 30 by coneill-math
- □ Parse fractions properly and implement GMP **bug** — #150 opened on May 23 by dkahle — 14