



SEEK: Science Environment
for Ecological Knowledge

Experiences in Integration of the 'R' System into Kepler

Dan Higgins – National Center for Ecological Analysis and
Synthesis (NCEAS), UC Santa Barbara

Prepared for Sixth Biennial Ptolemy Miniconference, May 12, 2005
at UC Berkeley

<http://seek.ecoinformatics.org>

<http://www.kepler-project.org>

This material is based upon work supported by the National Science Foundation under award 0225676.



What is 'R' ?

"R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R."

"R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, ...) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity."

From the R Project Web page - <http://www.r-project.org/>





Ptolemy/Kepler and R

- R language has many similarities to the PTH/Kepler expression language
- R language emphasizes operations on vectors, matrices, and tables (in R, 'data frames') rather than scalars. (This eliminates many explicit looping statements)
- Many detailed statistical operations and data manipulation routines already exist in R
- R has ability to create sophisticated graphic displays
- Being able to call R routines from Kepler would greatly simplify many workflows

R Example

```
C:\work\kepler\workflows\R/sample.dat
"DATE", "TIME", "T_AIR", "RH", "DEW", "BARO", "WD", "WS", "RAIN", "SOL", "SOL_SUM"
"01/01/01", "00:00", 15.0, 99, 14.5, 953.4, 099, 0.8, 0.0, 0.0000, 0.0000000
"01/01/01", "01:00", 13.4, 99, 12.8, 953.8, 100, 1.9, 0.0, 0.0000, 0.0000000
"01/01/01", "02:00", 13.4, 99, 12.8, 954.0, 114, 1.2, 0.0, 0.0000, 0.0000120
"01/01/01", "03:00",
"01/01/01", "04:00",
"01/01/01", "05:00",
"01/01/01", "06:00",
"01/01/01", "07:00",
"01/01/01", "08:00",
"01/01/01", "09:00",
"01/01/01", "10:00",
"01/01/01", "11:00",
"01/01/01", "12:00",
"01/01/01", "13:00",
"01/01/01", "14:00",
"01/01/01", "15:00",
"01/01/01", "16:00",
"01/01/01", "17:00",
"01/01/01", "18:00",
"01/01/01", "19:00",
"01/01/01", "20:00",
"01/01/01", "21:00",
"01/01/01", "22:00"

R Console
File Edit Misc Packages Help
> df <- read.table("C:/work/kepler/workflows/R/sample.dat", sep=",", header=TRUE)
> pairs(df)
> summary(df)
```

DATE	TIME	T_AIR	RH	DEW	BARO	WD	WS	RAIN	SOL	SOL_SUM
01/01/01	00:00	15.0	99	14.5	953.4	099	0.8	0.0	0.0000	0.0000000
01/01/01	01:00	13.4	99	12.8	953.8	100	1.9	0.0	0.0000	0.0000000
01/01/01	02:00	13.4	99	12.8	954.0	114	1.2	0.0	0.0000	0.0000120
01/01/01	03:00									
01/01/01	04:00									
01/01/01	05:00									
01/01/01	06:00									
01/01/01	07:00									
01/01/01	08:00									
01/01/01	09:00									
01/01/01	10:00									
01/01/01	11:00									
01/01/01	12:00									
01/01/01	13:00									
01/01/01	14:00									
01/01/01	15:00									
01/01/01	16:00									
01/01/01	17:00									
01/01/01	18:00									
01/01/01	19:00									
01/01/01	20:00									
01/01/01	21:00									
01/01/01	22:00									

DATE	TIME	T_AIR	RH	DEW	BARO	WD	WS	RAIN	SOL	SOL_SUM
01/01/01	00:00	5	Min.	: 8.90	Min					
01/02/01	01:00	5	1st Qu.	:12.20	1st					
01/03/01	02:00	5	Median	:15.15	Med					
01/04/01	03:00	5	Mean	:16.06	Mea					
01/05/01	04:00	4	3rd Qu.	:20.15	3rd					
01/05/01	05:00	4	Max.	:24.40	Max					
(Other): 72										

BARO	WD	WS
Min. :950.2	Min. : 2.00	Min. :0.000
1st Qu.:952.0	1st Qu.: 96.75	1st Qu.:0.300
Median :953.5	Median :113.50	Median :1.000
Mean :953.2	Mean :157.43	Mean :1.335
3rd Qu.:954.4	3rd Qu.:230.25	3rd Qu.:2.300
Max. :955.5	Max. :360.00	Max. :4.600

With only 3 lines, one can read a data table, plot all combinations of column data, and summarize the data



Interactive R in Kepler

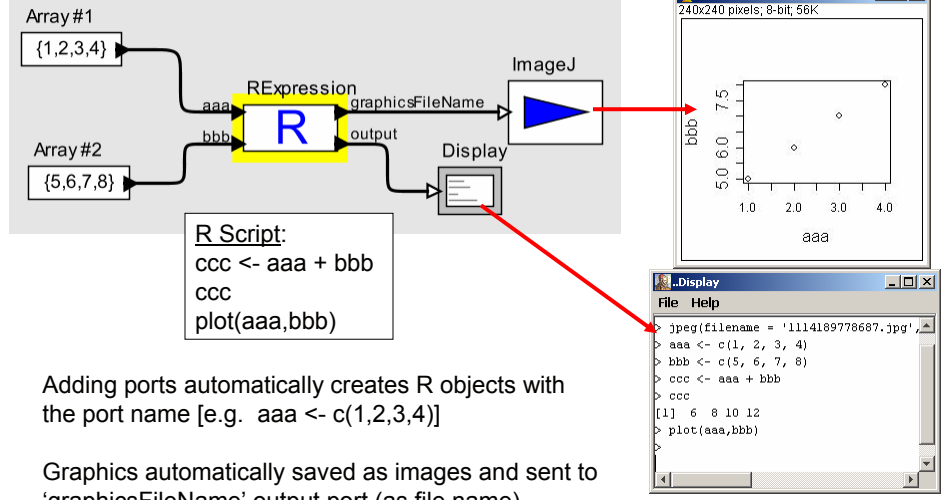


Efforts to R in Kepler

- First Effort --- Interactive R actor
 - No real advantage over existing R console
- Use of Command Line Actor
 - Problems: R initialization
 - How to get data in/out ? (files)
 - How to display graphics ? (files)
- RExpression actor
 - Use concepts from Kepler/PT Expression language/actor
- Using RServer



RExpression Actor



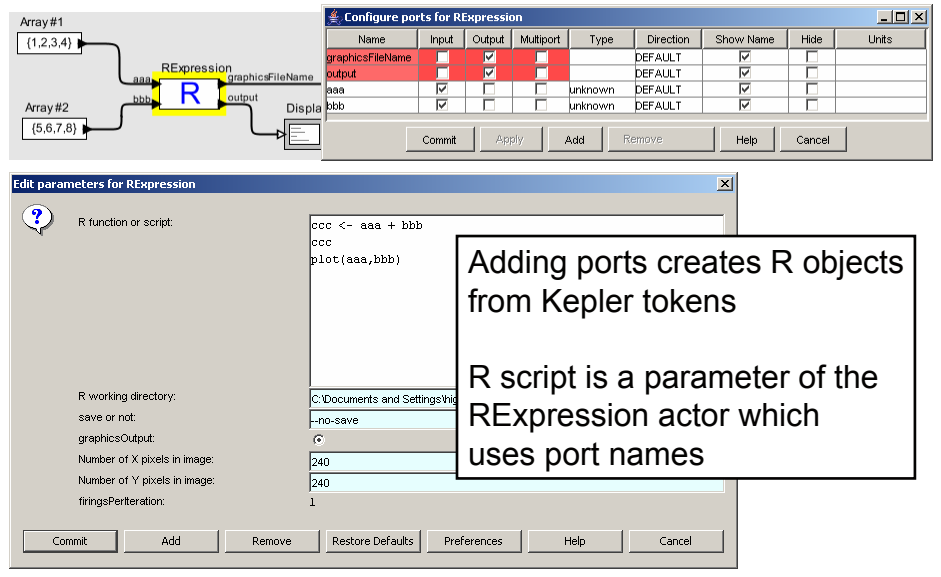
Adding ports automatically creates R objects with the port name [e.g. `aaa <- c(1,2,3,4)`]

Graphics automatically saved as images and sent to 'graphicsFileName' output port (as file name)

R text output automatically sent to 'output' port



RExpression – Ports & Parameters



Adding ports creates R objects from Kepler tokens

R script is a parameter of the RExpression actor which uses port names

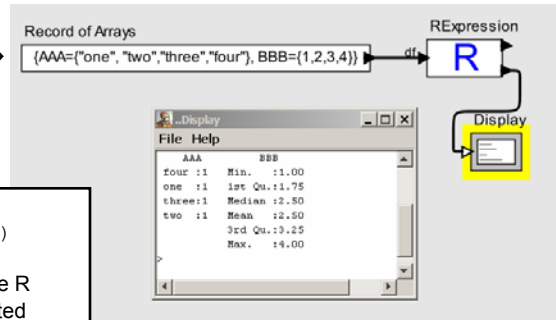


Array Records and Data Frames

Tables are represented as 'Data Frame' objects in 'R'

A Ptolemy 'Record of Arrays' can also represent a table

AAA	BBB
one	1
two	2
three	3
four	4



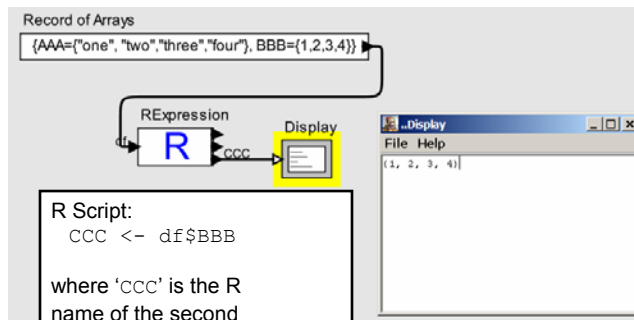
R Script:
summary(df)

where 'df' is the R dataframe created automatically when a record of arrays is passed to an input port



RExpression Output Ports

R vectors can also be assigned to output ports

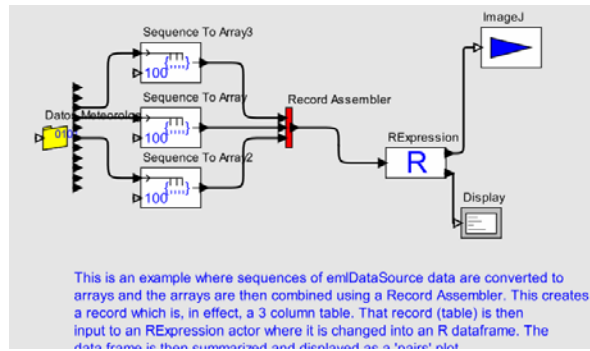


R Script:
CCC <- df\$BBB

where 'ccc' is the R name of the second column of the dataframe



EML DataSource Sequence Inputs



EML DataSource actor provides table data from SEEK Ecogrid

Column data from table can be supplied in various ways

Sequences of tokens from EML DataSource can be converted to arrays and then to a Record for input to RExpression



EML DataSource as Column Record

EML DataSource can be configured to create a "Column Based Record" directly for input to RExpression

File Help			
SOL	SOL_SUM	TIME	T_AIR
Min. : 0.0	Min. : 0	00:00 : 5	Min. : 0.90
1st Qu.: 0.0	1st Qu.: 0	01:00 : 5	1st Qu.:12.20
Median : 0.0	Median : 360	02:00 : 5	Median :15.15
Mean :258.0	Mean : 930382	03:00 : 5	Mean :16.06
3rd Qu.:564.5	3rd Qu.:1984590	04:00 : 4	3rd Qu.:20.15
Max. :982.0	Max. :3558000	05:00 : 4	Max. :24.40
		(Other):72	
WS		WS	
Min. : 2.00	Min. :10.000		
1st Qu.: 96.75	1st Qu.:10.300		
Median :113.50	Median :1.000		
Mean :157.43	Mean :1.335		
3rd Qu.:230.25	3rd Qu.:2.300		



R Regression Analysis Example

The diagram shows a workflow starting with a 'Data Meteorologicos' input. This data is processed by 'Sequence To Array2' and 'Sequence To Array3' blocks, which then feed into an 'R' block. The 'R' block outputs to 'ImageJ' and 'Display'.

This is an example of how one can carry out a simple linear regression analysis using R and add the regression line to a scatter plot.
Dan Higgins - March 2005

```
> T_AIR <- c(15.0, 13.4, 13.4, 12.4, 11.7, 11.4, 11.5, 11.5, 12.2, 17.4, 20.1)
> BARO <- c(953.4, 953.8, 954.0, 954.3, 954.5, 954.7, 954.8, 954.8, 954.9, 955.0)
> res <- lm(BARO ~ T_AIR)
> res

Call:
lm(formula = BARO ~ T_AIR)

Coefficients:
(Intercept)      T_AIR
 958.3772      -0.3244

> plot(T_AIR, BARO)
> abline(res)
```



R Summarize Table By Species

The diagram shows a workflow starting with a 'Mollusc population abundance monitoring: Fall 2000 mid-marsh and creekbank infaunal and epifaunal' input. This data is processed by an 'R' block, which then outputs to 'ImageJ' and 'Display'.

```
summary(maa[[1]])
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00000 0.00000 0.00000 0.01266 0.00000 1.00000

summary(maa[[2]])
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000 0.0000 0.0000 0.2125 0.0000 4.0000
```



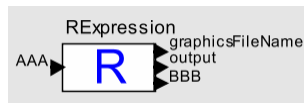
RExpression Implementation - 1

1. Input Ports

Kepler tokens are converted to R string expressions
e.g. If port AAA has token {1,2,3} it is converted
to the R expression '`AAA <- c(1,2,3)`'
Automatically handles strings, numbers, arrays, and
records with arrays of the same length

2. R Command Line Process

R is started as a Java subprocess with text streams
attached to standard in, out, and error



RExpression Implementation - 2

3. Input Block of R Commands

A set of R commands are sent to the input stream of
the R subprocess

Initialization

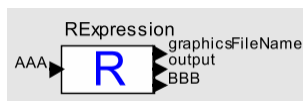
Create graphics device (jpeg file);
Create input port objects

User Script

Whatever is in user's script
`BBB <- 2 * AAA`

Finalization

R commands for output ports (e.g. BBB)



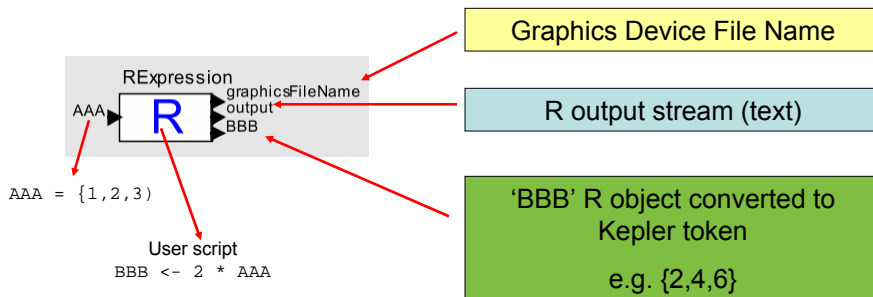


RExpression Implementation - 3

4. Execute R

Send input block to R subprocess and get output

5. Put R results on appropriate output ports



RServe

“**Rserve** is a TCP/IP server which allows other programs to use facilities of **R** without the need to initialize **R** or link against **R** library.”

Client-side implementations are available for C/C++ and Java.

-----Java code example -----

```
Rconnection c = new Rconnection();  
double d[]=c.eval("rnorm(10)").asDoubleArray();  
-----
```

Use of RServe would avoid each actor 're-starting' R and allow remote execution of R scripts

RServe --- <http://stats.math.uni-augsburg.de/Rserve/>



Summary

An RExpression actor that operates similarly to the existing Expression actor looks like a good way of integrating R into Kepler

Using R in Kepler provides powerful extensions to the Ptolemy expression language that allows operations on complex structures (e.g. tables)

Existing implementation is inefficient in some ways and incomplete, but is relatively easy to use and does not require detailed knowledge of R for simple operations