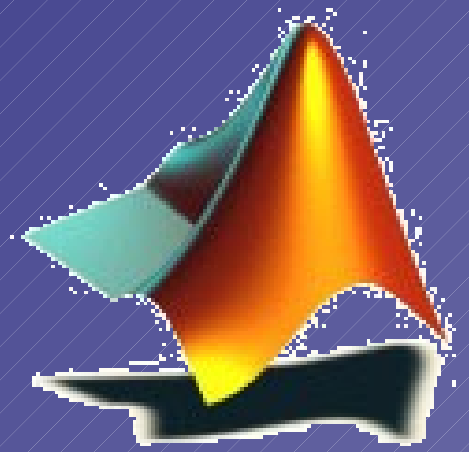
MATLAB Basics

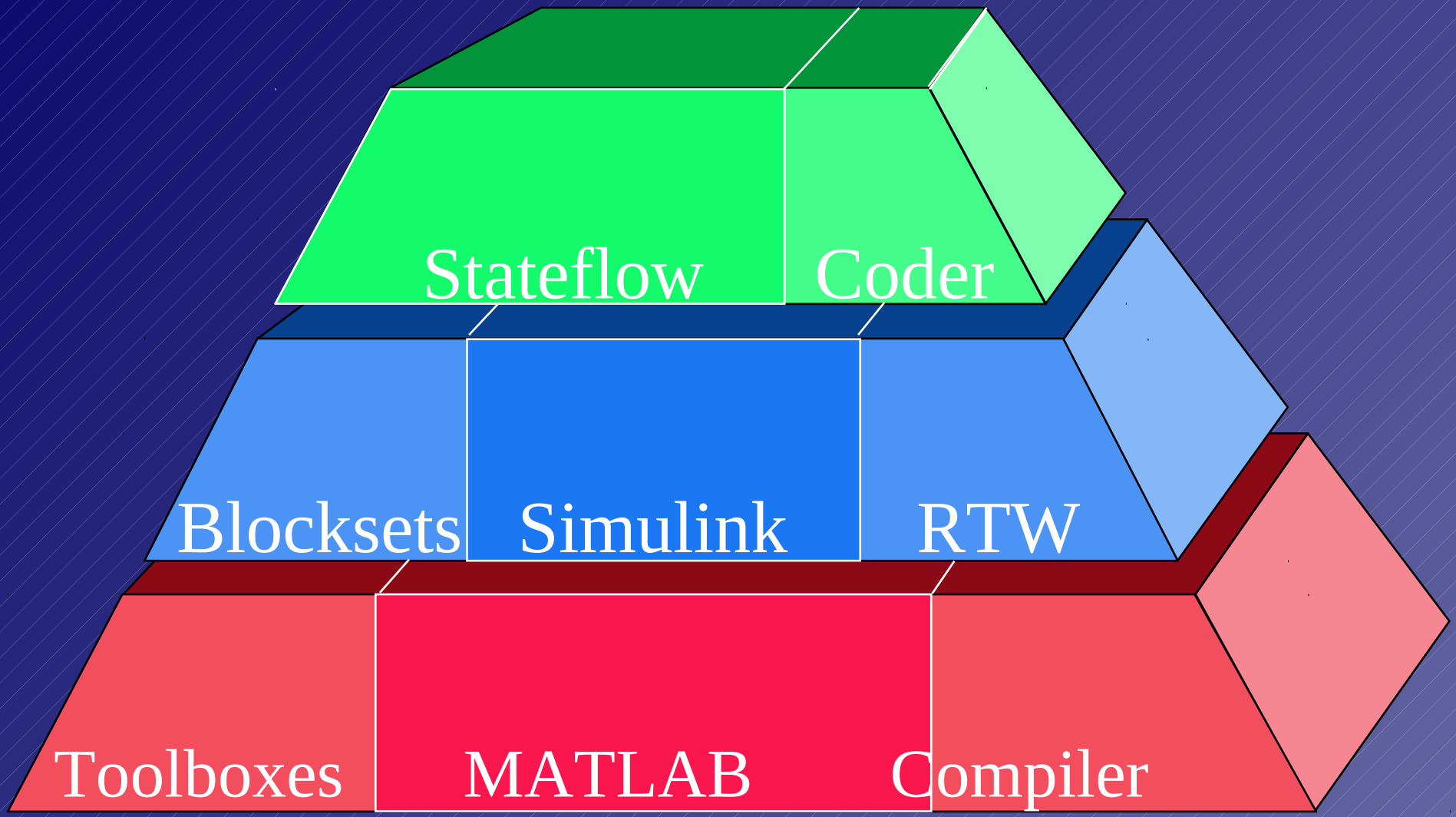
Barak Shenhav

Early Evolution Course

28 Jan 2001



The MathWorks Product Suite



What Is MATLAB ?

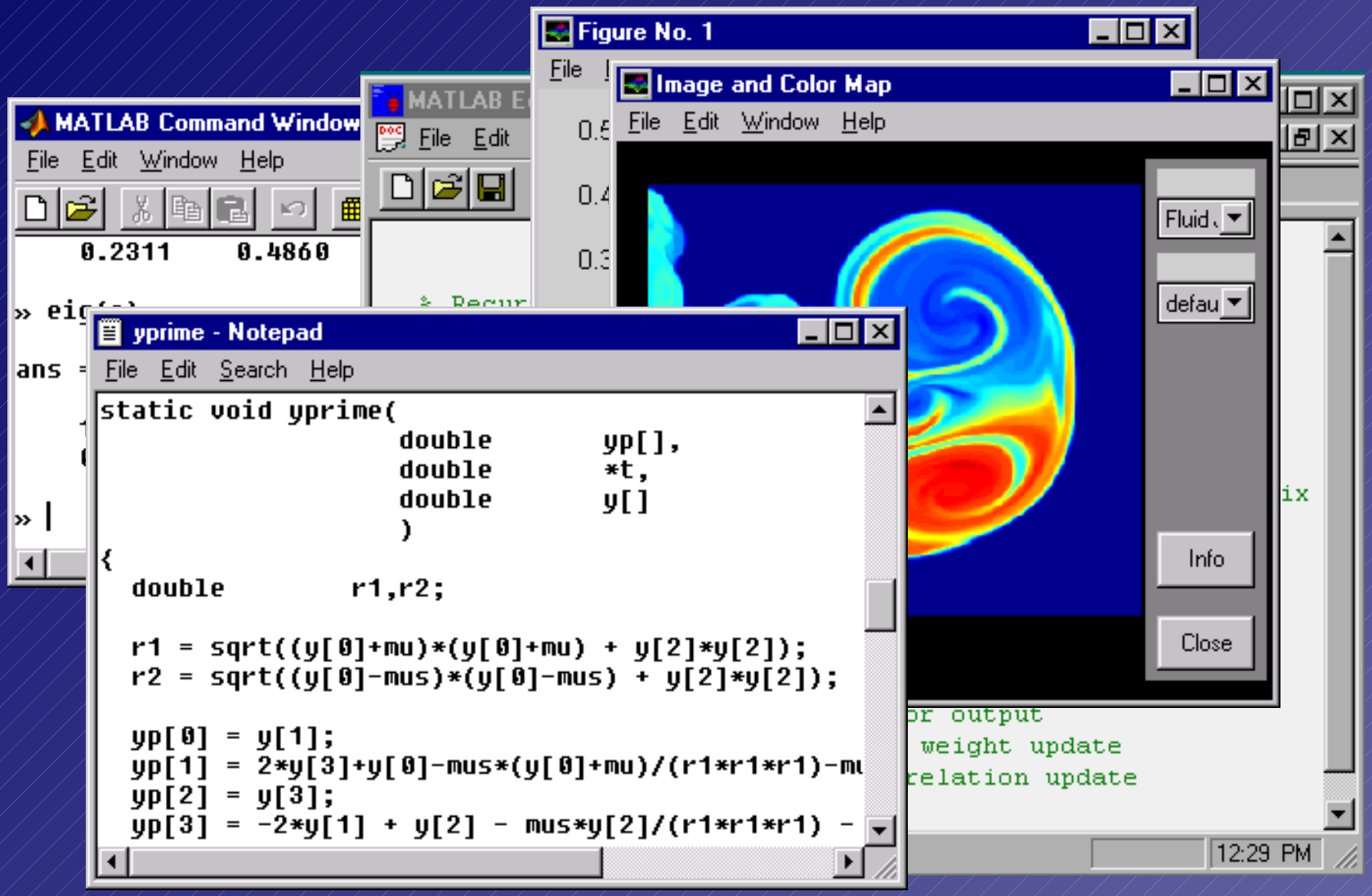
MATLAB is a high-performance language for technical computing.

It integrates :

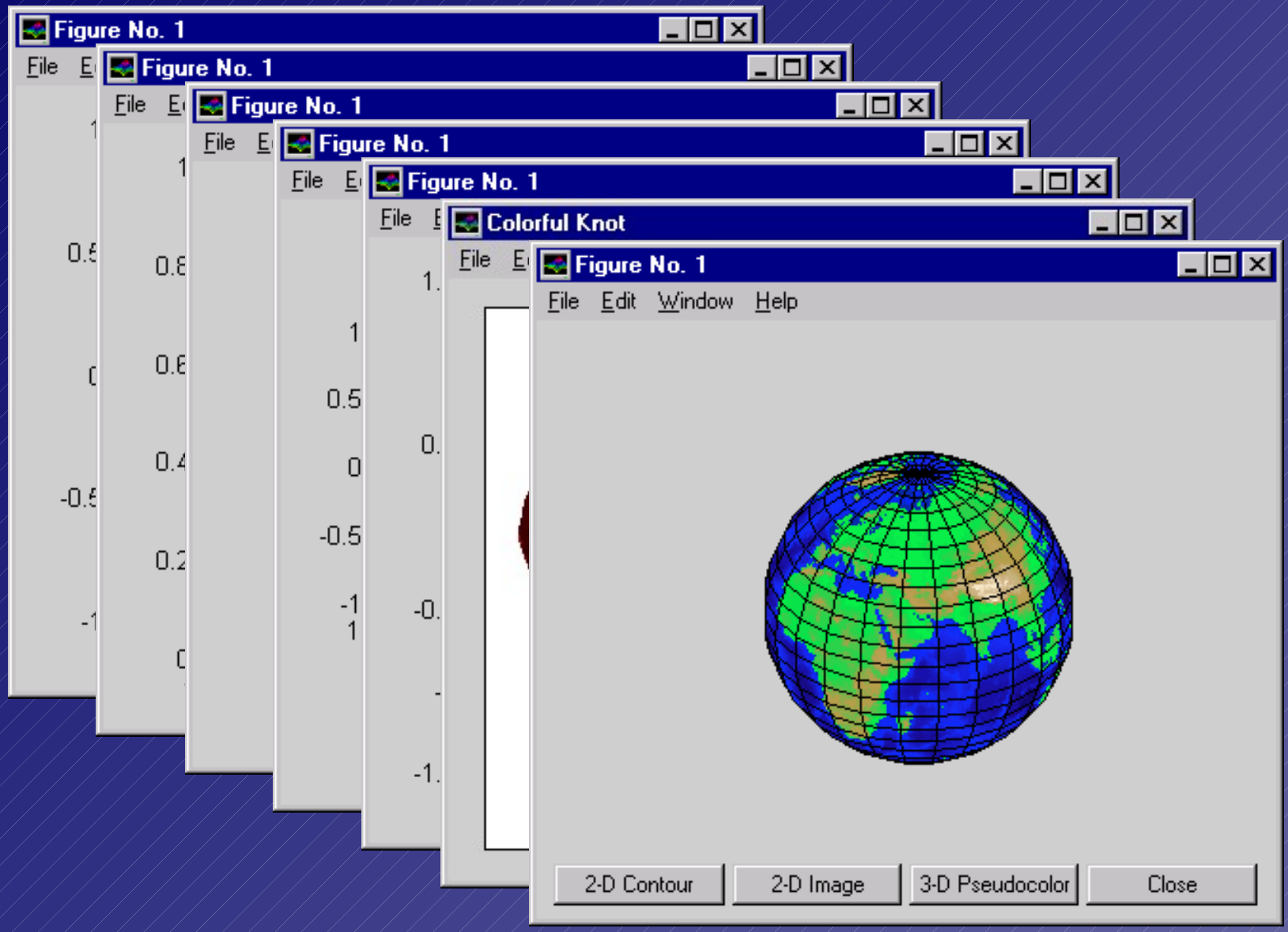
- computation
- visualization
- programming

in an easy-to-use environment

MATLAB Components

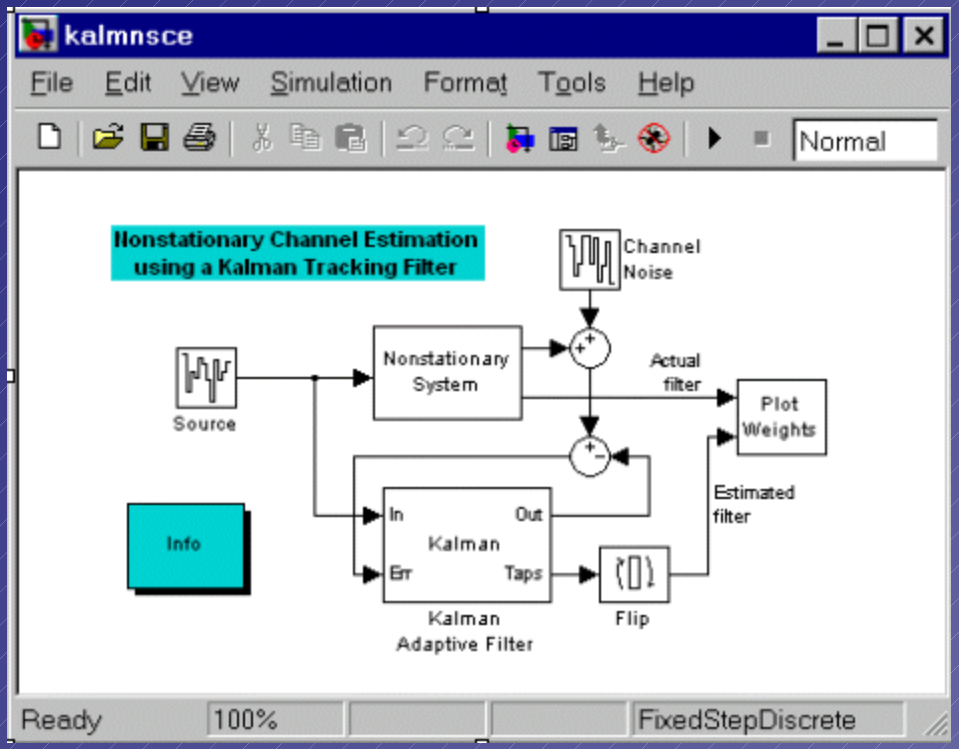


Visualization

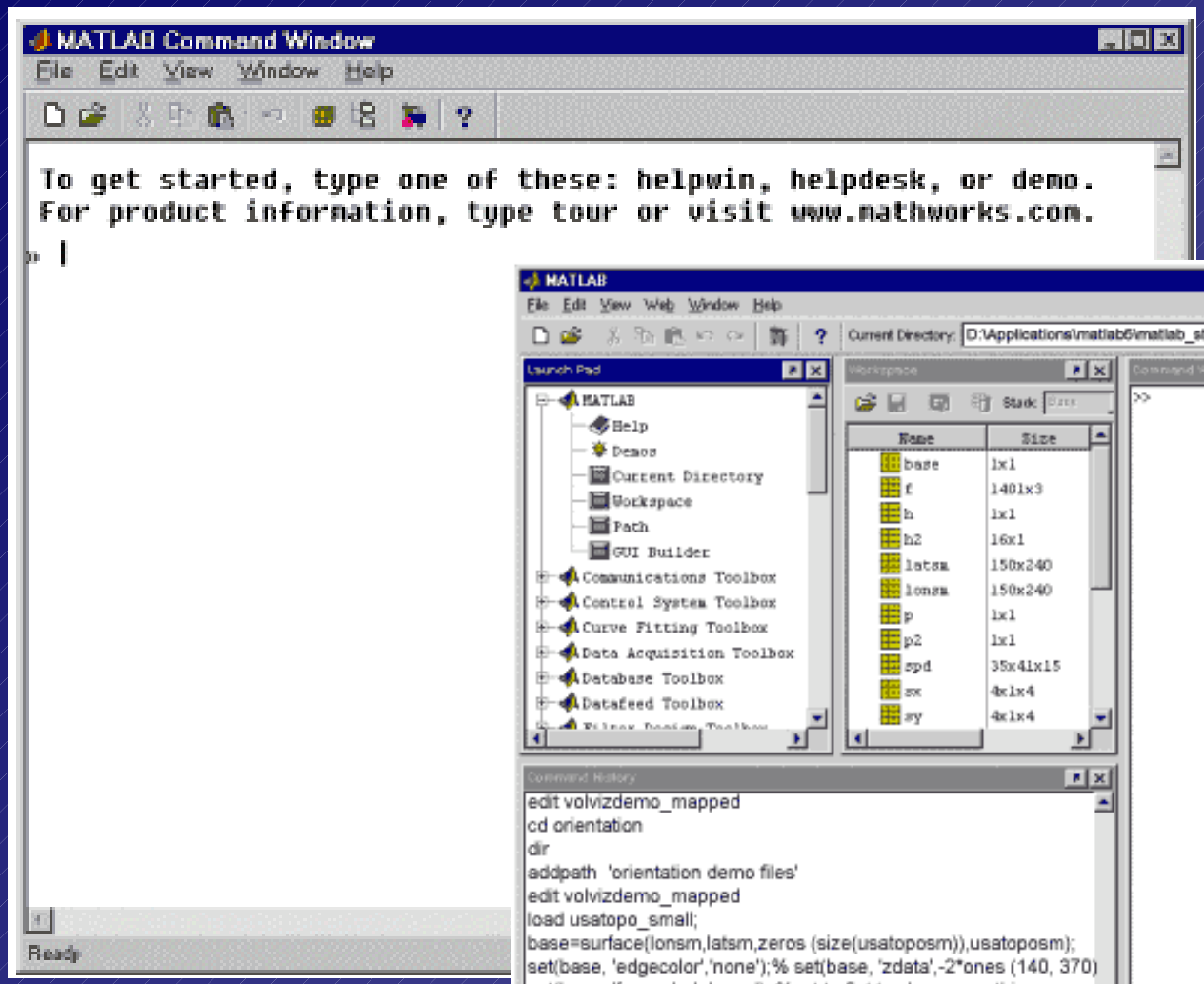


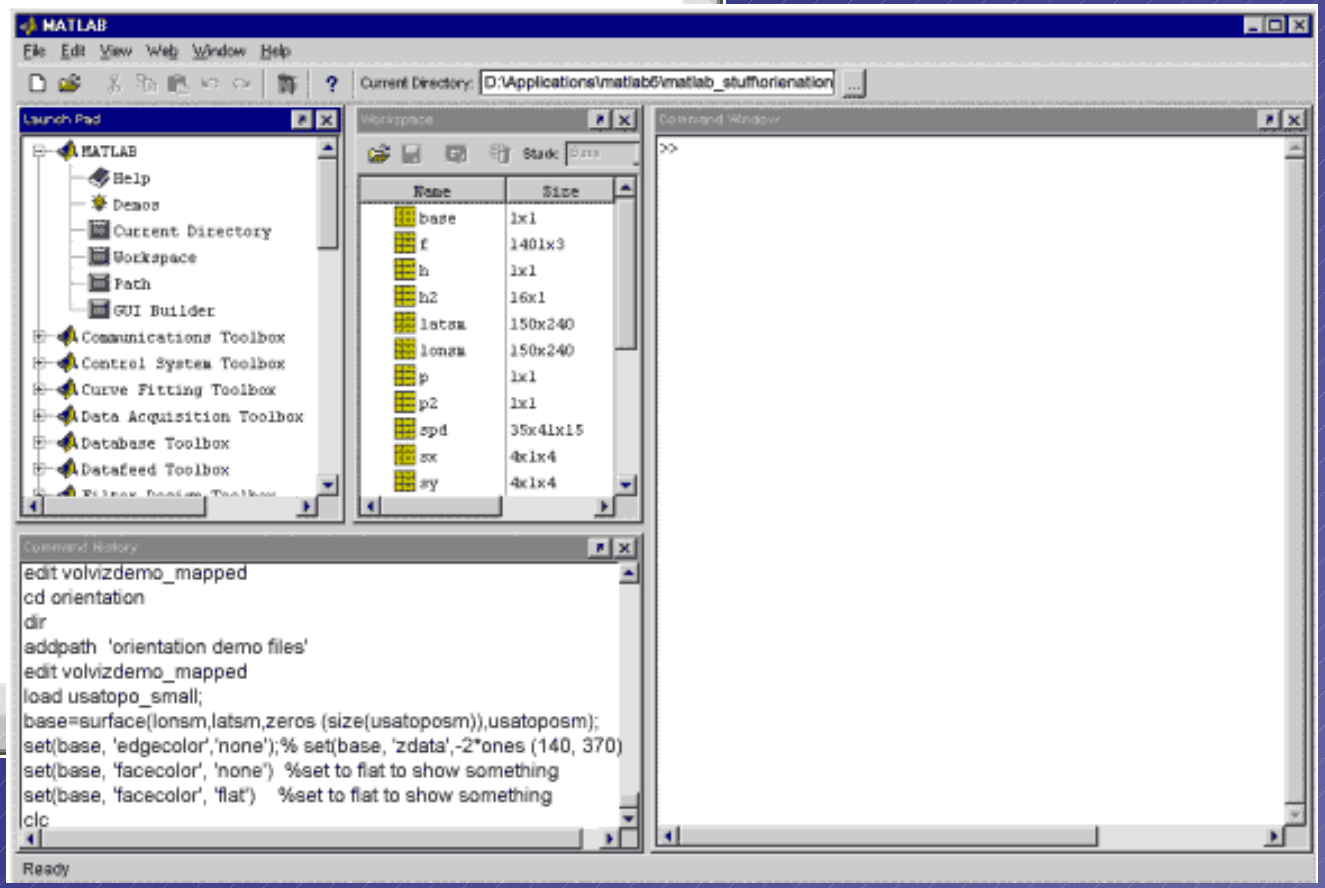
About Simulink

Simulink is a graphical, mouse-driven, interactive system for simulating dynamic systems. It allows you to model a system by drawing it's block diagram using library Blocksets or costume made blocks



Getting Started





Matrices

Entering Matrices:

- Enter an explicit list of elements.
- Load matrices from external data files.
- Generate matrices using built-in functions.
- Create matrices with your functions (m-files).

A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]

- Separate the elements of a row with blanks or commas.
- Use a semicolon, ; , to indicate the end of each row.
- Surround the entire list of elements with square brackets, [ ].

MATLAB stands for matrix laboratory.

Magic ?

```
A =
    16     3     2    13
     5    10    11     8
     9     6     7    12
     4    15    14     1
```

sum(A)

A`  ==>  sum(A`)`

diag(A) ==> sum(diag(A))

fliplr(A) ==> sum(diag(fliplr(A)))

The *magic* Function

B = magic(4)

```
B =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

The Colon (:) Operator

1:10 ==> [1 2 3 4 5 6 7 8 9 10]

0:10:50 ==> [0 10 20 30 40 50]

0:pi/4:pi ==> [0 0.7854 1.5708 2.3562 3.1416]

Subscripts

```
A =
    16     3     2    13
     5    10    11     8
     9     6     7    12
     4    15    14     1
```

The element in row i and column j of A is denoted by A(i,j), for example A(4,2) is 15.

The i^{th} row is denoted by A(i,:) and the j^{th} column is denoted by A(:,j), for exemple A(3,:) is [9 6 7 12].

Expressions

Like most other programming languages, MATLAB provides mathematical expressions, but unlike most programming languages, these expressions involve entire matrices.

The building blocks of expressions are

- Numbers
- Variables
- Functions
- Operators

Numbers

MATLAB uses conventional decimal notation for numbers. For example:

3, -99, 0.0001, 3.14159, 1.60210e-20, 6.02252e23, 1i, 1j 3e5i

All numbers are stored using the long format specified by the IEEE floating-point standard (i.e. 8 bytes). Floating-point numbers have a finite precision of roughly 16 significant decimal digits and a finite range of roughly 10-308 to 10+308.

Variables

When MATLAB encounters a new variable name, it automatically creates the variable and allocates the appropriate amount of storage. MATLAB does not require any type declarations or dimension statements.

Variable names consist of a letter, followed by any number of letters, digits, or underscores. MATLAB uses only the first 31 characters of a variable name. MATLAB is case sensitive (i.e. A and a are not the same variable).

Functions

MATLAB provides a large number of standard elementary mathematical functions, such as: abs, sqrt, exp, and sin. MATLAB handles complex numbers thus taking the square root or logarithm of a negative number is not an error.

For a list of the elementary functions:

help elfun

help specfun

help elmat

Operators

Expressions use familiar arithmetic operators and precedence rules.

+	Addition
-	Subtraction
*	Multiplication
/	Division
\	Left division
^	Power
`	Complex conjugate transpose
( )	Specify evaluation order

Expressions Examples

```
rho = (1+sqrt(5))/2      ==>  rho = 1.6180
a = abs(3+4i)            ==>  a = 5
z = sqrt(-1)             ==>  z = 0 + 1i
huge = realmax           ==>  huge = 1.7977e+308
toobig = pi*huge         ==>  toobig = Inf
invnumber = 0/0          ==>  invnumber = NaN
a = exp(log(100))        ==>  a = 100
```

Generating Matrices

MATLAB provides four functions that generate basic matrices:

zeros	All zeros
ones	All ones
rand	Uniformly distributed random elements
randn	Normally distributed random elements

```
R = randn(3,5)

R =

    1.0668    0.2944   -0.6918   -1.4410    0.0593
   -1.3362    0.8580    0.5711   -0.0956    0.7143
    1.2540   -0.3999   -0.8323    1.6236   -1.5937
```

The Command Window

- Adding a semicolon (;) at the end of a line suppresses output. This is particularly useful when you generate large matrices.
- If a statement does not fit on one line, use three periods, ..., followed by Return or Enter to indicate that the statement continues on the next line. For example:
  s = 1 -1/2 + 1/3 -1/4 + 1/5 - 1/6 + 1/7 …
  - 1/8 + 1/9 - 1/10 + 1/11 - 1/12;
- Various arrow and control keys on your keyboard allow you to recall, edit, and reuse commands you have typed earlier.

Scripts

Scripts are text files containing MATLAB code. MATLAB script files should have the extension .m

For example, if the file magik.m contains:

```
C = [3 5 7; 4 9 2; 8 1 6];
sum(C)
```

than typing magik in the MATLAB prompt will produce the output:

```
ans =
     15     15     15
```

Flow Control

MATLAB has five flow control constructs:

- if statements
- switch statements
- for loops
- while loops
- break statements

If

The if statement evaluates a logical expression and executes a group of statements when the expression is true. The optional elseif and else keywords provide for the execution of alternate groups of statements. An end keyword, which matches the if, terminates the last group of statements.

```
 if rem(n,2) ~= 0
    M = odd_magic(n)
elseif rem(n,4) ~= 0
    M = single_even_magic(n)
else
    M = double_even_magic(n)
end
```

Logical Expressions

Relational operators: ==, ~=, >, <, <=, >=
Logical operators: & (and), | (or), ~ (not)
Logical functions: isequal, isempty, all, any

for example:

```
a1 = ones(2, 3);
a2 = [1 1 1; 1 1 1];
b = [];
c = 6;
if (isequal(a1, a2) | isempty(b)) & all(0:10 > 0.5) & c,
    disp('TRUE')
end
```

Switch and Case

The switch statement executes groups of statements based on the value of a variable or expression. The keywords case and otherwise delineate the groups. Only the first matching case is executed. There must be an end to match the switch.

```
switch (rem(n,4)==0) + (rem(n,2)==0)
   case 0
      M = odd_magic(n)
   case 1
      M = single_even_magic(n)
   case 2
      M = double_even_magic(n)
   otherwise
      error('This is impossible')
end
```

For

The for loop repeats a group of statements a fixed, predetermined number of times. A matching end delineates the statements.

```
r = zeros(32, 1)
for n = 3:32
   r(n) = rank(magic(n));
end
r
```

While

The while loop repeats a group of statements an indefinite number of times under control of a logical condition. A matching end delineates the statements.

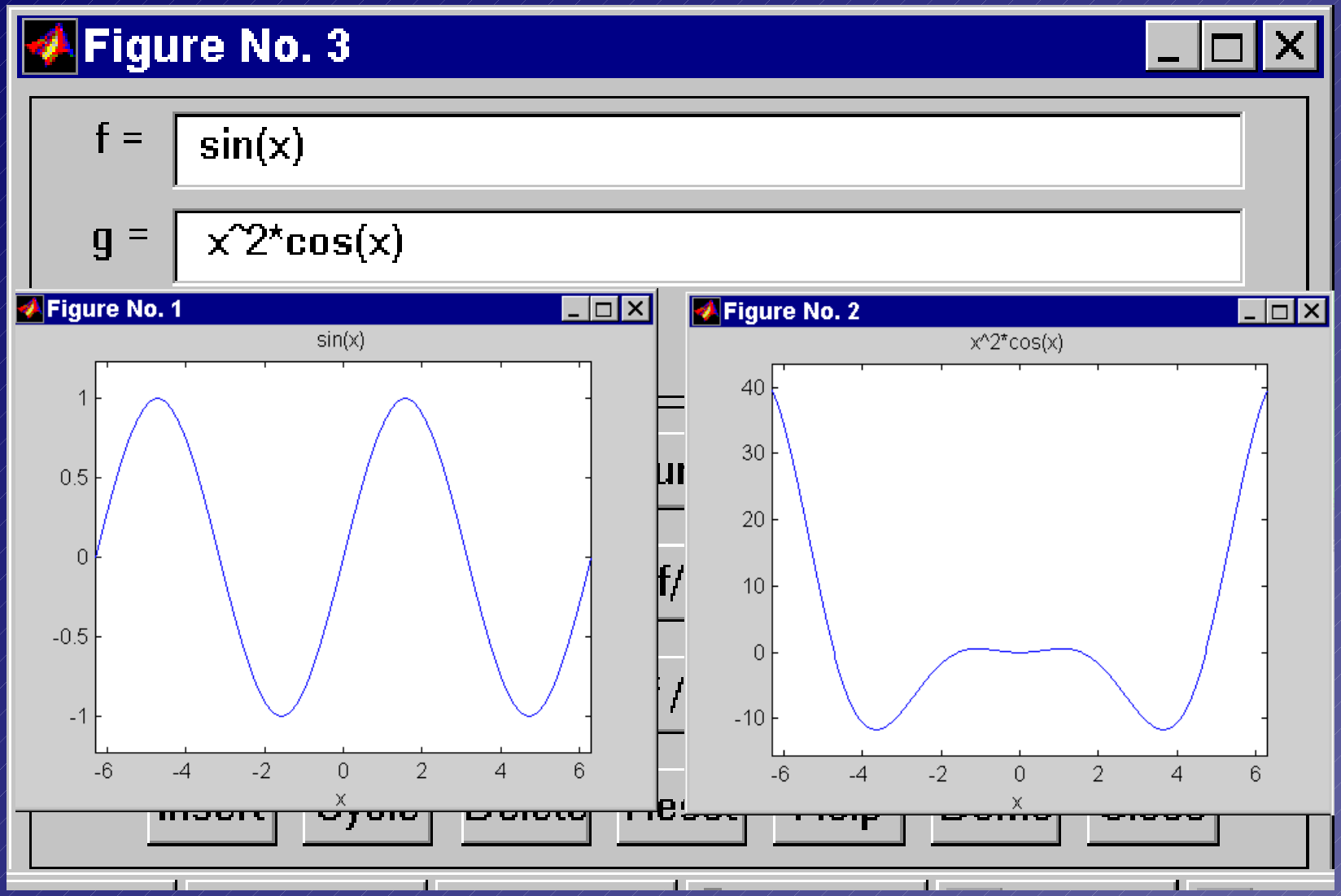
```
a = 0; fa = 8;
b = 3; fb = -4;
while b-a > eps + b
   x = (a+b)/2;
   fx = x^3 - 5x^2 + 2x + 8;
   if sign(fx) == sign(fa)
      a = x; fa = fx;
   else
      b = x; fb = fx;
   end
end
x
```

Break

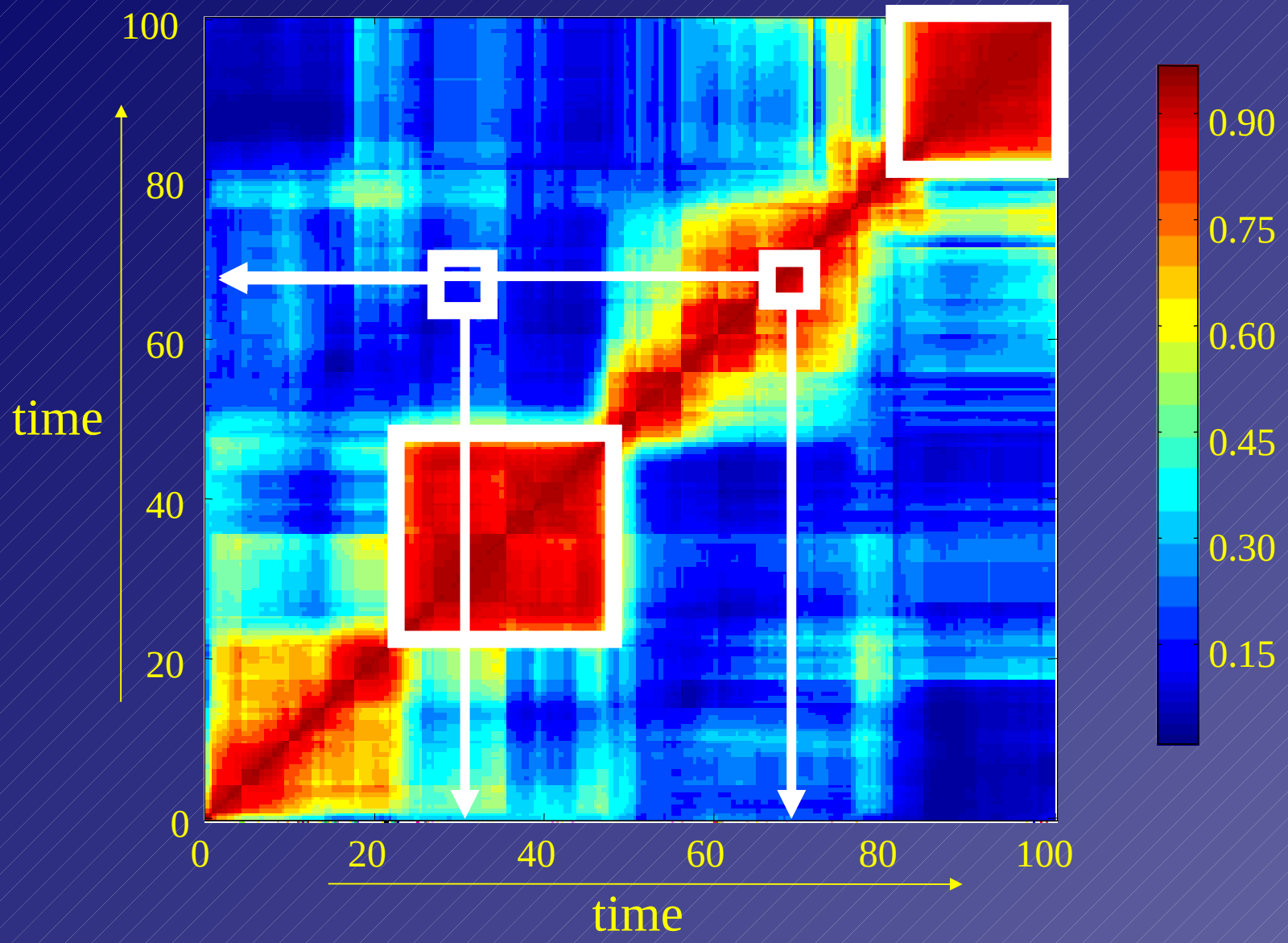
The break statement lets you exit early from a for or while loop. In nested loops, break exits from the innermost loop only.

```
a = 0; fa = 8;
    b = 3; fb = -4;
    while b-a > eps + b
        x = (a+b)/2;
        fx = x^3 - 5x^2 + 2x + 8;
        if fx == 0
            break;
        elseif sign(fx) == sign(fa)
            a = x; fa = fx;
        else
            b = x; fb = fx;
        end
    end
    x
```

Symbolic Math Toolbox-Funtool



Compositional Carpet



Heritability – Fidelity of Replication

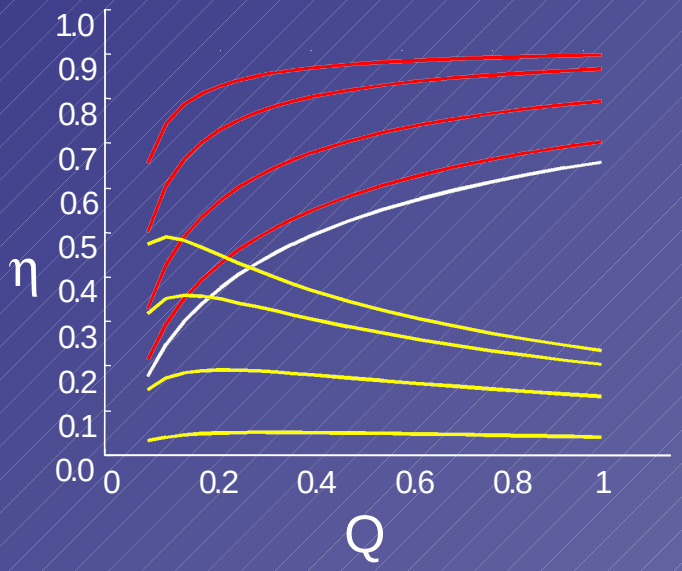
A measure for the fidelity of self-replication

$$\eta(\mathbf{n}) = \langle H(\mathbf{n}, \mathbf{n}`) \rangle_{\mathbf{n}`} = \sum_{\mathbf{n}`} H(\mathbf{n}, \mathbf{n}`) \cdot P(\mathbf{n}`|\mathbf{n})$$

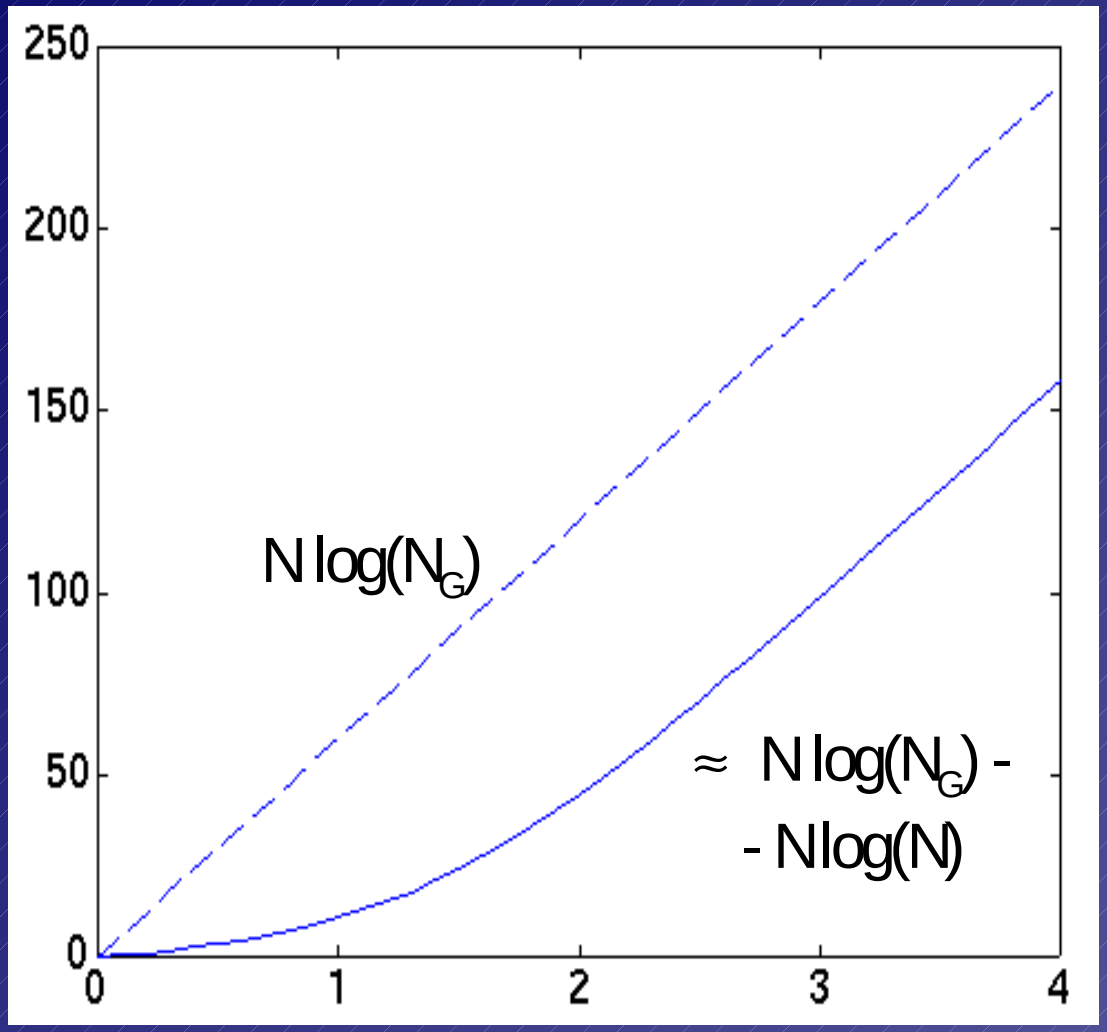
$\eta^* = \eta(\mathbf{n}^*)$ Heritability

$\eta_B = \eta(^{\text{Equi-}}_{\text{molar}})$ Trivial heritability

$\eta_S^* = \eta^* - \eta_B$ Non-trivial, specific heritability



Sequence vs. Compositional Information

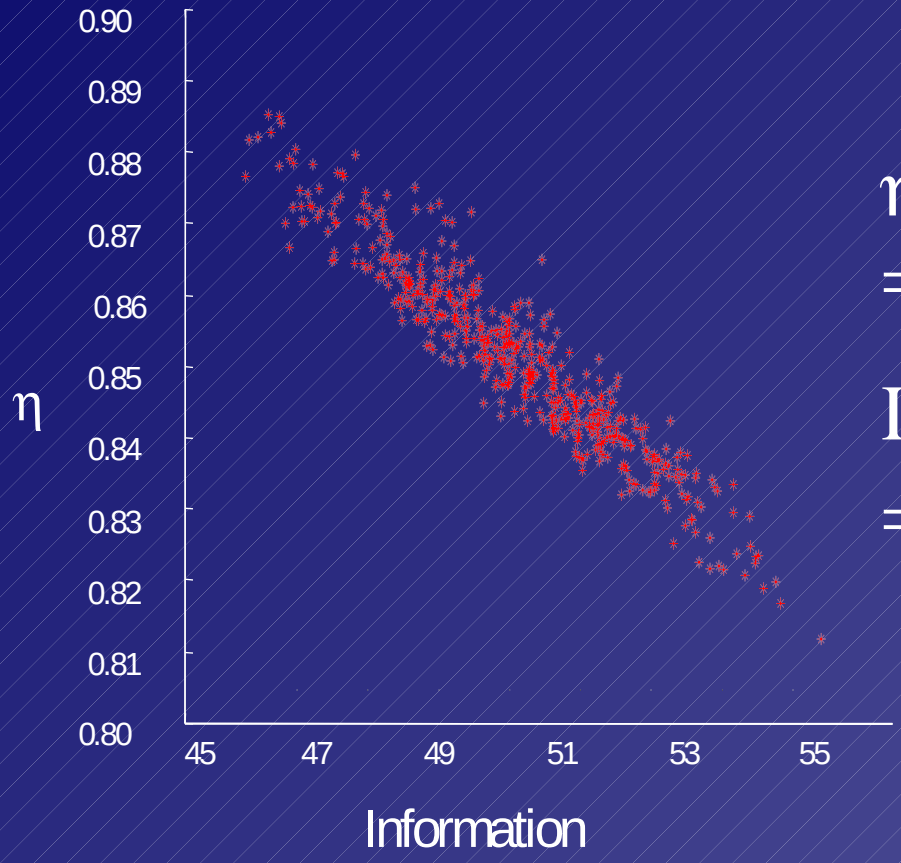


Compositions -

$$\frac{(N_G + N - 1)!}{(N_G - 1)!\, N!}$$

polymers - N_G^N

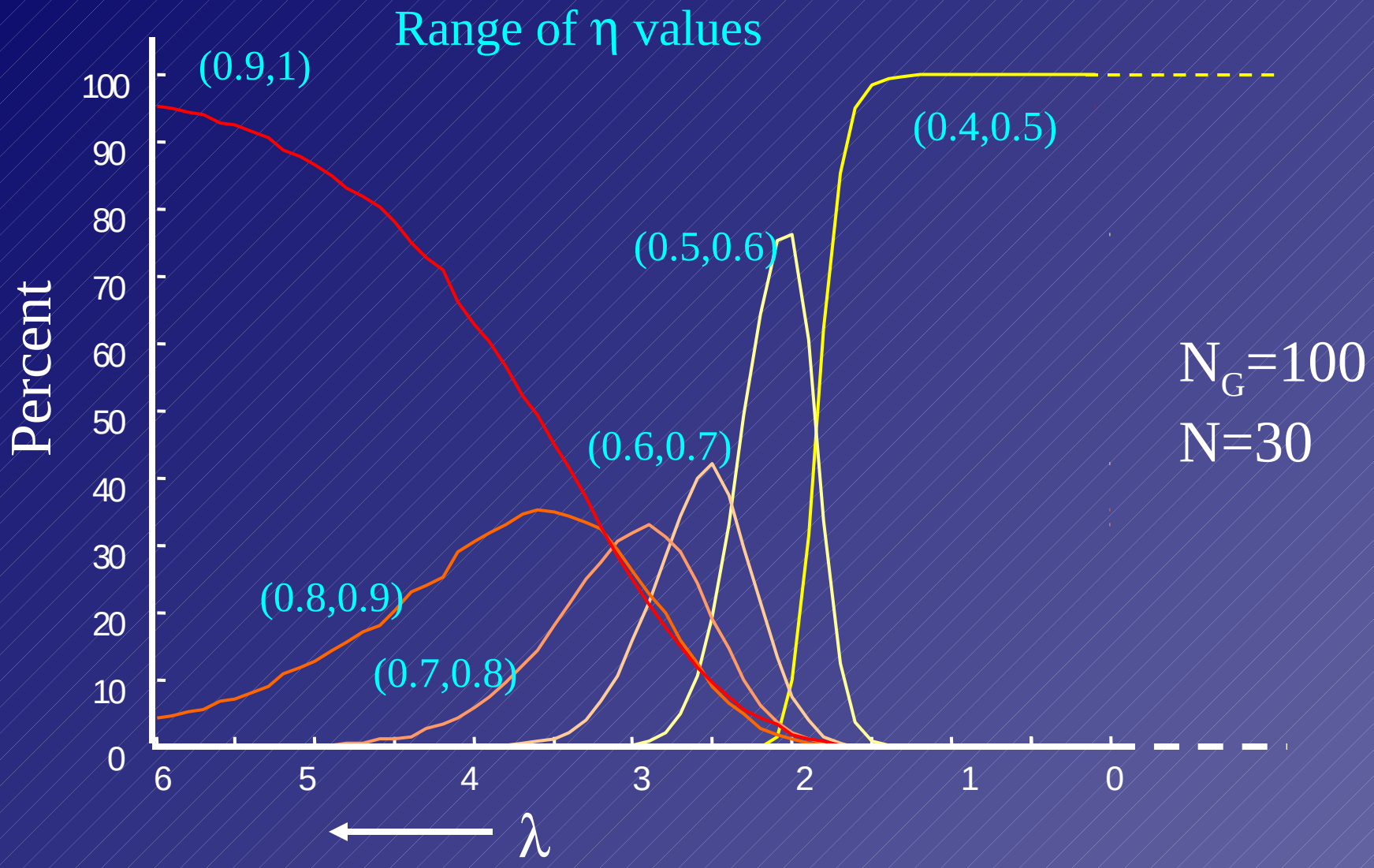
Heritability and Shanon's Information



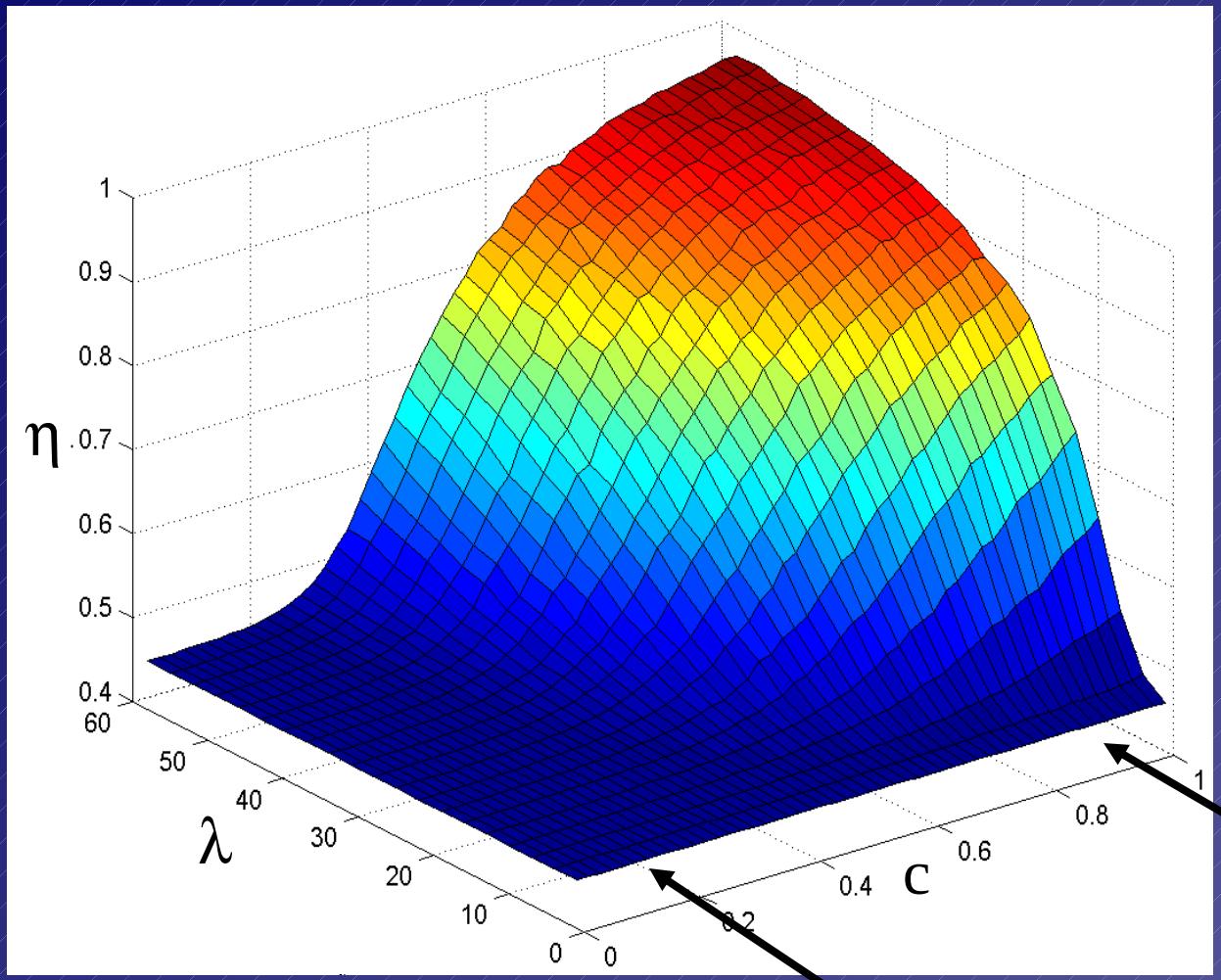
$$\eta(\mathbf{n}) = \sum_{\mathbf{n}`} P(\mathbf{n}`|\mathbf{n}) \cdot H(\mathbf{n}, \mathbf{n}`)$$

$$I_S(\mathbf{n}) = \sum_{\mathbf{n}`} P(\mathbf{n}`|\mathbf{n}) \cdot \log P(\mathbf{n}`|\mathbf{n})$$

Compositional Error Catastrophe



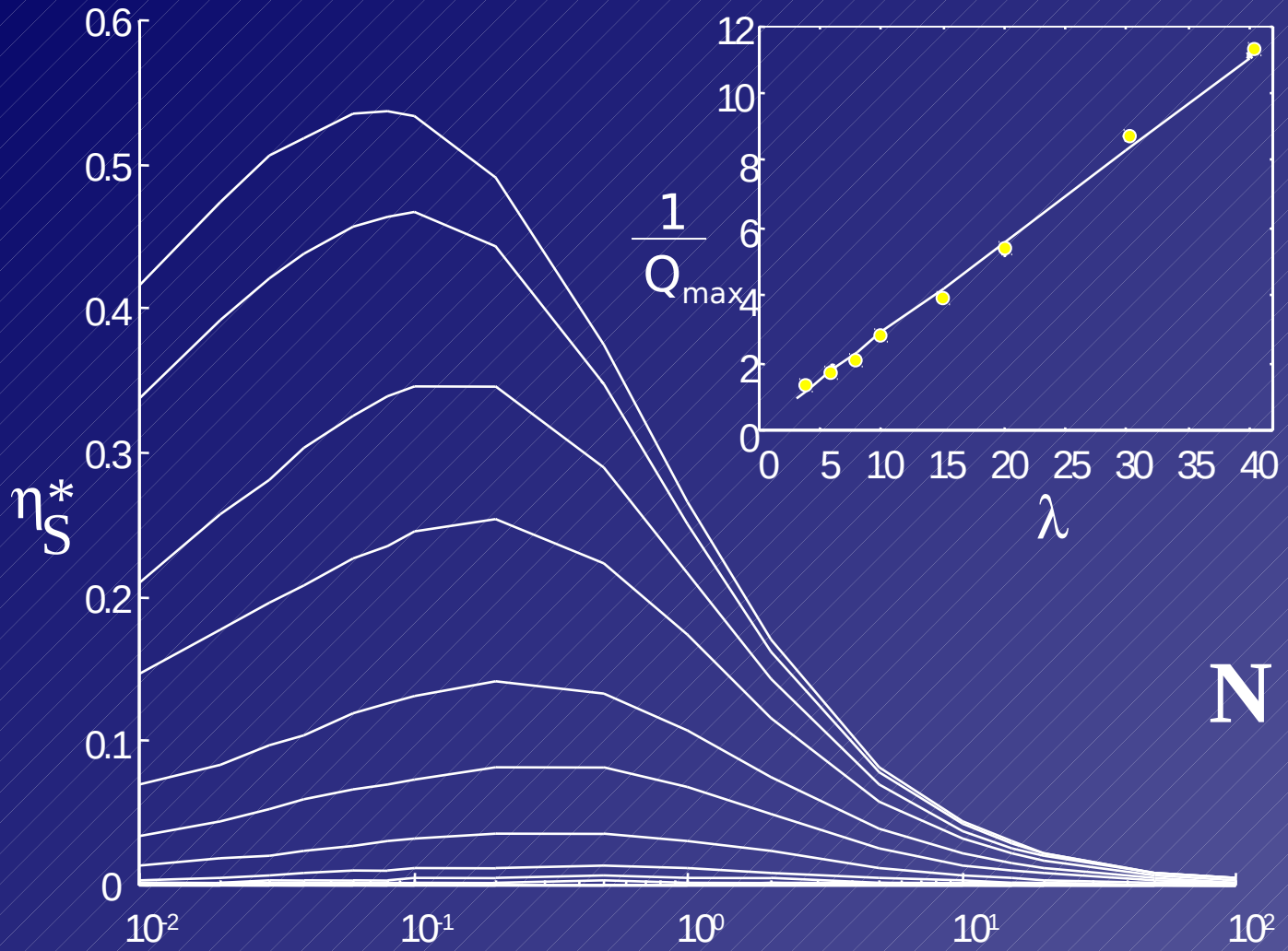
The Importance of Being Lognormal



Lognormal

Normal

Optimal Assembly Size



$$Q = \frac{N}{N_G}$$

$$N \sim \frac{3.5N_G}{\lambda}$$

Where Did Life Start ?

Life Started Here