Benefits of Ada for Scientific Computing

David Smith
Lunar and Planetary Laboratory
What is Ada?

- Designed for critical systems where bugs could kill people: avionics, spacecraft, nukes, etc.

  Why shouldn't scientific programming be the same?


- First standardized object-oriented language -- not C++!
Ada's Namesake

Augusta Ada, Countess of Lovelace (1815-1852)
Hello, world!

with Ada.Text_IO; use Ada.Text_IO;

procedure Hello is
begin
  Put_Line("Hello, world!");
end Hello;
Strong Typing

- Eliminates confusion about variable conversion
  \[ \text{izone} = 0.5 \times (1 + 10) \quad -- \quad 5 \text{ or } 6? \]

- No implicit variables.

- No uninitialized variables.

- Assignments between variables of different types require an explicit type conversion.
Float Sweetness

To help in debugging and code clarity, you can:

- Require decimal digits of precision:
  ```
  type Real is digits 15;
  ```

- Constrain values:
  ```
  Angle : Float range -Pi .. Pi;
  ```

Features not present in C or Fortran.
Array Sweetness

- Sizes can be determined at run time.
- Element operations:  \( A := B + C; \)
- Array slicing:  \( A(1 .. 10) := B(11 .. 20); \)
- Compiler can distinguish different array types and indices are arbitrary.

\[
\begin{align*}
\text{Float}_{-}\text{Array}(1 .. 10) \\
\text{Float}_{-}\text{Array}(0 .. 9) & \quad \text{-- not the same!} \\
\text{Float}_{-}\text{Array}(-10 .. -1)
\end{align*}
\]
Attributes

• Over 90 attributes accessible with the ' operator.

```plaintext
X : array (1 .. 10) of Float;

X'Range          -- range of indices (1..10)
X'First           -- first valid index (1)
X'Last            -- last valid index (10)
Float'Model_Small -- smallest float
Float'Max(x,y)    -- max of x and y
```
English-like Constructs

for I in A'Range loop
    A(I) := A(I) + 1;
end loop;

while X > 0 loop
    X := X - 1;
end loop;
Helpful Syntax for Scientists

- Embedded underscores in constants.

\[
\text{Pi} : \text{constant} := 3.14159_26535_89793_\ldots
\]

- ** for exponentiation.

- \((i,j)\) for array indexing.

- Ada recognizes “wide” characters, so Greek symbols can be used for math.

\[
\Sigma := \frac{M}{(\pi \times R^{**2})}; \quad -- \text{legal!}
\]
Compile- and Run-Time Checks

- Using variables before initialization.
- Array index out of bounds.
- Assignment to variable violates specified constraints.
  
  ```
  Theta : Float range -Pi .. Pi;
  Theta := 2.0 * Pi; -- Illegal!
  ```

- Incorrect arguments to math functions.
Loop Sweetness

• Loops use automatic dummy variables.

• Loops can be labeled for deep exit and clarity.

```plaintext
Outer_Loop:
  for I in Arr'Range loop
    for J in 1 .. 10 loop
      Arr(I) := Float(I+J);
      exit Outer_Loop when I + J > 10;
    end loop;
  end loop Outer_Loop;
```
Subprogram Sweetness

- **Parameter input/output specification:**

  ```plaintext
  procedure Quadratic (A, B, C : in Float; X1, X2 : in out Float);
  ```

- **Polymorphism**

  ```plaintext
  procedure Sort (A : in out Integer_Array);
  procedure Sort (A : in out Real_Array);
  ```
More Subprogram Sweetness

• Default and optional parameters

```plaintext
function Arctan (X : in Float; Y : in Float := 1.0)
return Float;

Arctan(X/Y)   -- atan
Arctan(X,Y)   -- atan2
```

• Argument labels can be used for clarity.

```plaintext
S := Integrate(Function => f,
                From  => a,
                To    => b,
                Error => 1e-5);
```
Packages

• Similar to Fortran 90 modules – can group variables, subroutines, etc., for import to other programs.

• Public and private data and subroutines.

• Separate compilation – no recompiling if nothing changed.
Tasks

• Subprogram that is executed in parallel.
• Can synchronize and communicate with main program.
• Tasks can have private data and also access global data.
Ada Efficiency and Speed

- Specifies more at compile time than C or Fortran, easing optimization.
- Built-in parallelization through tasks.
- Run-time checks can be turned off if desired.
- As fast as C and Fortran in benchmarks.
Interfacing with Other Languages

- Ada can interface with other languages, such as Fortran, C, and Cobol.
- Ada understands their data types and can call their subroutines.
- Existing numerical analysis libraries can be used: BLAS, LAPACK, etc.
Summary

• Ada is designed to reduce bugs.
• Ada is fast and efficient.
• Ada can interface with C and Fortran.
• Ada is syntactically convenient for science and math.

Ada makes a great scientific computing language.
Resources

- Ada 95 Lovelace Tutorial
  http://www.dwheeler.com/lovelace/lovelace.htm

- Ada for C programmers
  http://www.adahome.com/Ammo/cpp2ada.html

- Comparing Development Costs
  http://www.adaic.com/whyada/ada-vs-c/cada_art.html

- Programming in Ada 2005 by John Barnes