Numerical Representation of Planetary Ephemerides

X. X. Newhall, Celestial Mechanics 45:305-310, 1989

Computations

A handy function to compute the derivative of a Chebyshev polynomial.

In[24]:=
```
DChebyshevT = Derivative[0, 1][ChebyshevT]
```

Out[24]=
```
ChebyshevU[-1 + #1, #2] #1 &
```

This function computes matrix *T* from Newhall's equation (5). The parameter degree is the degree of the polynomial (*N* in Newhall), the parameter divisions is the number of subintervals of [-1, 1] (8 in Newhall).

In[25]:=
```
NewhallT[degree_Integer, divisions_Integer] :=
 Flatten[
  Table[
   {Table[ChebyshevT[j, i], {j, 0, degree}], Table[DChebyshevT[j, i], {j, 0, degree}]},
   {i, 1, -1, -2 / divisions}],
  {1, 2}]
```

This function computes matrix *W* used in Newhall's equation (8). The parameter *w* is the weight of the velocities relative to the positions (0.4 in Newhall).

In[26]:=
```
NewhallW[divisions_Integer, w_Rational] :=
 DiagonalMatrix[Flatten[Table[{1, w^2}, {divisions + 1}]]]
```

The following functions compute the four blocks of matrix C1 and assemble them to form C1.

In[27]:=
```
NewhallC1UpperLeft[degree_Integer, divisions_Integer, w_Rational] :=
 NewhallT[degree, divisions]^T.NewhallW[divisions, w].NewhallT[degree, divisions]
```

In[28]:=
```
NewhallC1UpperRight[degree_Integer] :=
 Table[{ChebyshevT[i, 1], DChebyshevT[i, 1],
   ChebyshevT[i, -1], DChebyshevT[i, -1]}, {i, 0, degree}]
```

In[29]:=
```
NewhallC1LowerLeft[degree_Integer] := NewhallC1UpperRight[degree]^T
```

In[30]:=
```
NewhallC1LowerRight[] := Table[0, {4}, {4}]
```

In[31]:=
```
NewhallC1[degree_Integer, divisions_Integer, w_Rational] :=
 ArrayFlatten[
  {{NewhallC1UpperLeft[degree, divisions, w], NewhallC1UpperRight[degree]},
   {NewhallC1LowerLeft[degree], NewhallC1LowerRight[]}}]
```

The following functions compute the two blocs of matrix C_2 and assemble them to form C_2.

In[32]:=
```
NewhallC2Upper[degree_Integer, divisions_Integer, w_Rational] :=
 NewhallT[degree, divisions]^T.NewhallW[divisions, w]
```

In[33]:=
```
NewhallC2Lower[divisions_Integer] :=
 Drop[IdentityMatrix[2 divisions + 2], {3, 2 divisions}]
```

In[34]:=
```
NewhallC2[degree_Integer, divisions_Integer, w_Rational] :=
 ArrayFlatten[{{NewhallC2Upper[degree, divisions, w]}, {NewhallC2Lower[divisions]}}]
```

This function computes the matrix $C_1^{-1}.C_2$. Newhall doesn't give it a name but calls its elements c_k, so let's use the name C.

In[35]:=
```
NewhallC[degree_Integer, divisions_Integer, w_Rational] :=
 Inverse[NewhallC1[degree, divisions, w]].NewhallC2[degree, divisions, w]
```

This function expresses C in a way that is suitable for obtaining the coefficients of a polynomial in the monomial base, not in the Chebyshev base. It drops the last 4 rows corresponding to the Lagrange multipliers.

In[36]:=
```
NewhallMonomialC[degree_Integer, divisions_Integer, w_Rational] :=
 Table[
  Sum[
   NewhallC[degree, divisions, w][[n]] Coefficient[ChebyshevT[n - 1, x], x, k],
   {n, 1, degree + 1}],
  {k, 0, degree}
 ]
```

Formatting and Output

Produces a representation of a matrix as an initializer_list containing initializer_lists. (Note that this function is unused and might need to change, e.g., to use std::array if we wanted to use it.)

In[37]:=
```
BidimMatrixToCDefinition[type_String, variable_String, matrix_List] :=
 type <> "  const\r\n     " <> variable <> "(\r\n" <>
  StringReplace[
   ToString[CForm[matrix]],
   {"List(List(" → "         {{",
    "List(" → "{",
    ")," → "},\r\n         ",
    "," → ",\r\n          ",
    "))" → "}});\r\n\r\n"}]
```

Produces a representation of a matrix as a single, flattened initializer list.

In[38]:=
```
FlattenedMatrixToCDefinition[type_String, element_String,
  dimension1_String, dimension2_String, variable_String, matrix_List] :=
 "constexpr " <> type <> "<" <> element <> ", " <> dimension1 <> ", " <>
  dimension2 <> ">\r\n     " <> variable <> "(\r\n         std::array<" <>
  element <> ", " <> "(" <> dimension1 <> ") * (" <> dimension2 <> ")>{\r\n" <>
  StringReplace[
   ToString[CForm[matrix]],
   {"List(List(" → "             {",
    "List(" → "\r\n              ",
    ")," → ",\r\n",
    "," → ",\r\n               ",
    "))" → "}});\r\n\r\n"}]
```

Produces a representation of a list as an initializer list.

In[39]:=
```
ListToCDefinition[type_String, variable_String, list_List] :=
 type <> "  const\r\n     " <> variable <> "(\r\n" <>
  StringReplace[
   ToString[CForm[list]],
   {"List(" → "         {",
    "," → ",\r\n           ",
    ")" → "});\r\n\r\n"}]
```

Writes all the Newhall *C* matrices to a single file. Note that we drop the last 4 rows because they correspond to the Lagrange multipliers.

In[40]:=
```
file =
  OpenWrite[
   FileNameJoin[{DirectoryName[NotebookDirectory[]], "numerics",
     "newhall.mathematica.h"}], BinaryFormat → True, PageWidth → Infinity];
WriteString[
  file,
  FromCharacterCode[16^^ef] <>
   FromCharacterCode[16^^bb] <> FromCharacterCode[16^^bf] <>
   "// Generated by Mathematica.  DO NOT EDIT!\r\n",
  "// source: mathematica/newhall.nb\r\n",
  "\r\n",
  "#include <array>\r\n",
  "\r\n",
  "#include \"numerics/fixed_arrays.hpp\"\r\n",
  "\r\n",
  "namespace principia {\r\n",
  "namespace numerics {\r\n", "\r\n"];
Do[
  WriteString[
   file,
   FlattenedMatrixToCDefinition[
    "FixedMatrix", "double", ToString[degree] <> " + 1", "2 * 8 + 2",
    ToString["newhall_c_matrix_чебышёв_degree_", CharacterEncoding → "UTF8"] <>
     ToString[degree] <> "_divisions_8_w04",
    Drop[NewhallC[degree, 8, 4 / 10], -4]]];
  WriteString[
   file,
   FlattenedMatrixToCDefinition[
    "FixedMatrix", "double", ToString[degree] <> " + 1", "2 * 8 + 2",
    "newhall_c_matrix_monomial_degree_" <> ToString[degree] <> "_divisions_8_w04",
    Drop[NewhallC[degree, 8, 4 / 10], -4]],
  {degree, 3, 17}];
WriteString[
  file,
  "}  // namespace numerics\r\n",
  "}  // namespace principia\r\n"];
Close[file];
```

Save a pdf printout of this file for documentation purposes.

In[45]:=
```
printout =
  FileNameJoin[{DirectoryName[NotebookDirectory[]], "documentation", "newhall.pdf"}];
NotebookPrint[EvaluationNotebook[], printout]
```