SCCS2014 Model Validation: Resources

Martin Wood and Maike Sonnewald (ICSS&NOCS)

15th August 2014

Abstract

Many of the attendees reading this will be working between disciplines and often on abstract questions. Though we appreciate that we cannot begin to cover the vast number of datasets that now exist across the many scientific disciplines, we have below collated a few interesting, unusual or powerful datasets that may be of interest to the reader. This sheet makes for a nice launch point for those wishing to explore datasets relevant to their work. We hope you find this and the workshop to be useful! Please feel free to contact us if you have any questions. Naturally those who work in the relevant disciplines already know their own data sources best. If you know where to go already, well done! If you don't, the best advice is to go meet people!

1 Data

1.1 Natural

The Paleobiology Database, macrofossils occurrence and timing
paleobiodb.org
"The Paleobiology Database (PaleoDB) is a non-governmental, non-profit public resource for paleontological data. It has been organized and operated by a multi-disciplinary, multi-institutional, international group of paleobiological researchers. Its purpose is to provide global, collection-based occurrence and taxonomic data for organisms of all geological ages, as well data services to allow easy access to data for independent development of analytical tools, visualization software, and applications of all types. The Databases broader goal is to encourage and enable data-driven collaborative efforts that address large-scale paleobiological questions."

NEPTUNE (on CHRONOS), marine microfossils occurrence and timing
CHRONOS
"NEPTUNE is a relational database of microfossil occurrences reported in DSDP and ODP samples.NEPTUNE contains the occurrences of more than 9000 plankton species names (nannofossils, foraminifera, diatoms, and radiolarians) in Cenozoic and Mesozoic samples of more than 300 DSDP and ODP drillholes from all ocean basins."

NOAA - National Climatic Data Center, National Oceanic and Atmospheric Administration
National Climatic Data Center
"NCDC is the worlds largest provider of weather and climate data. Land-based, marine, model, radar, weather balloon, satellite, and paleoclimatic are just a few of the types of datasets available."

NOAA - Paleoclimatology datasets from the National Oceanic and Atmospheric Administration
National Climatic Data Center - Paleoclimatology
"These links provide access to descriptive information and explanatory notes, maps, searches, visualizations, and more. The data cover the globe, and while most span the last few millennia, some datasets extend back in time 100 million years. Most of the data are time series of geophysical or biological measurements and some include reconstructed climate variables such as temperature and precipitation."

AVISO+ Satellite Altimetry Data
AVISO+
"Aviso have been distributing altimetric data worldwide since 1992. Since that date, satellite altimetry has evolved in parallel with the user community and oceanography."

CMIP5 - Coupled Model Intercomparison Project
CMIP5

"CMIP5 is meant to provide a framework for coordinated climate change experiments . . . CMIP5 promotes a standard set of model simulations in order to: - evaluate how realistic the models are in simulating the recent past - provide projections of future climate change on two time scales, near term (out to about 2035) and long term (out to 2100 and beyond) - understand some of the factors responsible for differences in model projections, including quantifying some key feedbacks such as those involving clouds and the carbon cycle"

GAPMINDER - "Gapminder World shows the World's most important trends"
GAPMINDER
"[Wealth, health, CO2, HIV, mortality, and more] Gapminders work serves a purpose of filling a gap. There has been a market failure in distributing global data. A lot of people are interested in the data, but dont get access to it (and if they manage to access the data, they need to be advanced statisticians to analyze it). Gapminder wants to make data more accessible and easier to use for instant visual analysis. We believe decision makers, politicians as well as education at almost all levels lack adequate tools."

Trends, Rhythms and Aberrations in Global Climate, a paper that serves as an introduction and portal to palaeoclimate proxy data
Trends, Rhythms and Aberrations
"Since 65 million years ago (Ma), Earth's climate has undergone a significant and complex evolution, the finer details of which are now coming to light through investigations of deep-sea sediment cores. This evolution includes gradual trends of warming and cooling driven by tectonic processes on time scales of 105to 107 years, rhythmic or periodic cycles driven by orbital processes with 104- to 106-year cyclicity, and rare rapid aberrant shifts and extreme climate transients with durations of 103 to 105 years. Here, recent progress in defining the evolution of global climate over the Cenozoic Era is reviewed. We focus primarily on the periodic and anomalous components of variability over the early portion of this era, as constrained by the latest generation of deep-sea isotope records. We also consider how this improved perspective has led to the recognition of previously unforeseen mechanisms for altering climate."

1.2 Sociological

Avon Longitudinal Study of Parents and Children (ALSPAC)
ALSPAC
"The Avon Longitudinal Study of Parents and Children (ALSPAC) is a large scale longitudinal study of children born in Avon during the early 1990s and has been established as a resource for the study of genetic and environmental factors contributing to long term health and development."

English Longitudinal Study of Ageing
ELSA
"The English Longitudinal Study of Ageing (ELSA) study, which began in 2002 (though the sample was constructed from earlier data) is a longitudinal survey of ageing and quality of life among older people. It explores the dynamic relationships between health and functioning, social networks and participation, and economic position as people plan for, move into and progress beyond retirement. The study is funded jointly by UK government departments and the National Institute on Aging, in the USA."

UK Data Service ¡- the two above are part of this initiative
UK Data Service
"The UK Data Service is a comprehensive resource funded by the ESRC to support researchers, teachers and policymakers who depend on high-quality social and economic data. Here you will find a single point of access to a wide range of secondary data including large-scale government surveys, international macrodata, business microdata, qualitative studies and census data from 1971 to 2011."

Statistics at DECC - Department of Energy and Climate Change
DECC Statistics
"DECC publishes official statistics relating to energy, climate change, energy efficiency, fuel poverty, radioactive incidents and coal health."

2 Tom Anderson

For a very nice discussion on the benefit of "complexity" in models, the paper "Plankton functional type modelling: running before we can walk?" Anderson (2005) is also highly recommended!

3 Daisyworld

The model presented was based on Dyke *et al.* (2007), while McDonald-Gibson (2006) or Watson and Lovelock (1982) are also useful reference points. The code that generated the examples illustrated was implemented in C and is appended in 4.

4 Statistical methods

A host of information and tutorials on statistical methods and tests are available online, but if the reader prefers a more organised, paper-based, applied approach to the subject the book linked below is highly recommended.
Practical Statistics for Field Biology

References

Anderson, T.R. (2005). Plankton functional type modelling: running before we can walk? J. Plankton Res.. 27 (11): 1073-1081. doi: 10.1093/plankt/fbi076

Dyke J. G., McDonald-Gibson J., Di Paolo E., Harvey I. R. (2007). Increasing complexity increases stability in a self-regulating ecosystem. Proceedings of IXth European Conference on Artificial Life, ECAL 2007, 133-142, Springer, Berlin.

McDonald-Gibson, J.: Investigating Gaia: A new mechanism for regulation. Thesis for degree of Masters of Science of Evolutionary and Adaptive Systems, University of Sussex (2006).

Watson, A., Lovelock, J: Biological Homeostasis of the global environment - the parable of Daisyworld. Tellus B 35 (1982) pp 284-289.

Code

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include "/home/maike/Documents/COMP23/assignment2/random.h"

int MAXLINE=1000;      /* Maximum length of string */
int INDIVIDUALS_X=2000,INDIVIDUALS_Y=2; /* Number of individuals */
int FORCING=1;           /* Set type of forcing 0 = const, 1 = linear increase, 2 = Sinusoidal */
int CONST=100;           /* The value to which the external forcing will be increased */
int TIMESTEPS=100000;     /* The disceete timesteps */
double alpha=0.0025, betha=0.1;   /* alpha and betha determine the relative strengths of population effects
     and external perturbation */
double mew=0.3;        /* Mutation rate */
double gamma_=0.01;      /* Death rate */
/*double R = 50;*/         /* Initial resource value */
double lambda = 0.04;      /* The lambda parameter, providing a measure of the span of the R parabola */
/*int a_really_big_number = 2147483647;*/


/* function prototype, declaring what functions will be used and stuff.. */
double** Make2DDoubleArray(int arraySizeX, int arraySizeY);
double random_number_inrange(int LOW, int HIGH );
int tournament(double **daisies , double R, int step_ind);
double fitness(double A, double R);
void Free2DdoubleArray(double**theArray, int arraySizeX);

int main( void ){
  int step_ind, step_time;     /* Time stepping and individual stepping.. */
  double P, P_old, P_inc, time_p_inc, t_regulated = 0;       /* External forcing */
  double **daisies;      /* The daisies/individuals */
  double meanA=0, sum_theta=0;     /* Two allelles in gene */
  double R_sum = 0;      /* Initialise R_sum as initial R */
        long ltime = time(NULL);
        int stime = (unsigned)ltime/2;
  double R = 0;
  int winner, atypes;
  int *count_a_values;
  int n = 75;
  int nr_of_counts, countingA, reg_time, nr_of_A_values;
  FILE *debug_fitness;
  FILE *file;        /*pointing to the file apparently, file stuff from Iain*/
  FILE *reg;

        remove("debugging_fitness_v2.csv");
        remove("dasy_output.csv");
        remove("regulation_time.csv");


  debug_fitness = fopen("debugging_fitness_v2.csv", "w");

        file = fopen("dasy_output.csv", "w"); /* NB: THIS ONLY APPENDS STUFF TO THE FILE */
  reg = fopen("regulation_time.csv", "a+");

  count_a_values = ( int *) malloc ( sizeof ( int )* n );
        if ( count_a_values == NULL ) {
                printf ( " ERROR : Out of memory \n " );
    return 1;
        }
        else {
                printf ("Have allocated %d bytes for array.\n " ,
                (int) sizeof (int)* n);
        }

        srand(stime);

  /* Initialise INDIVIDUALS individuals in a 2D array. */
  daisies = Make2DDoubleArray(INDIVIDUALS_X+1, INDIVIDUALS_Y);
  for(step_ind = 0; step_ind <INDIVIDUALS_X;step_ind++){
    daisies[step_ind][0]= random_number_inrange(15, 85);
    daisies[step_ind][1]=random_number_inrange(-1, 1);
  }
```

```
73
    time_p_inc = (double) CONST/TIMESTEPS;
75  P = 0;
    /*for(mew = 0; mew <= 1; mew+=0.1){*/
77
    for (step_time = 0; step_time <= TIMESTEPS; step_time++){
79      /* calculate the external forcing */
        switch (FORCING){
81      case 0: /* 0 = const */
          P_old = P;
83        P = CONST/2.;
          P_inc = fabs(P_old-P);
85        break;
        case 1: /* 1 = linear increase */
87        P_old = P;
          P = P+time_p_inc;
89        P_inc = fabs(P_old-P);
          break;
91      case 2: /* 2 = Sinusoidal */
          P_old = P;
93        P = sin(step_time*3.14/50000)*100;
          P_inc = fabs(P_old-P);
95        break;
        default:
97        printf("We don't care about your age!\n");
          break;
99      }

101     if(R >= 15 && R <= 85){
          for (step_ind = 0; step_ind < INDIVIDUALS_X; step_ind++){ /* Sum over all daisies contributions to R
              */
103         if (daisies[step_ind][1]>0){
              R_sum = R_sum +1;
105         }
            else if (daisies[step_ind][1]<0){
107           R_sum = R_sum -1;
            }
109       }
        }
111
      R = R+(alpha*R_sum + betha*(P-R));
113

115     for (step_ind = 0; step_ind < INDIVIDUALS_X; step_ind++){
          if (random_number_inrange(0, 1) <= gamma_){
117         winner=tournament(daisies, R, step_ind);
            daisies[step_ind][0] = daisies[winner][0];
119         daisies[step_ind][1] = daisies[winner][1];

121         if (random_number_inrange(0, 1) < mew){

123           daisies[step_ind][0] = daisies[step_ind][0]+(gaussrand());/*random_number_inrange(-1, 1);*/
              if(daisies[step_ind][0]<15)daisies[step_ind][0]=15;
125           if(daisies[step_ind][0]>85)daisies[step_ind][0]=85;
            }
127         if (random_number_inrange(0, 1) < mew){
              daisies[step_ind][1] = daisies[step_ind][1]+(gaussrand()/20);/*random_number_inrange(-0.05, 0.05)
                  ;*/
129           if(daisies[step_ind][1]<-1)daisies[step_ind][1]=-1;
              if(daisies[step_ind][1]>1)daisies[step_ind][1]=1;
131         }
          }
133     }

135     for (step_ind = 0; step_ind < INDIVIDUALS_X; step_ind++){
          meanA = meanA + daisies[step_ind][0];
137       sum_theta = sum_theta + daisies[step_ind][1];
          /*printf("%7.3f, %7.3f, %7.3f\n", daisies[step_ind][0], daisies[step_ind][1], meanA/INDIVIDUALS_X);*/
139     }

141       for(countingA=0; countingA<70; countingA++){
            count_a_values[countingA]=0;
143     }

145     for(step_ind = 0; step_ind < INDIVIDUALS_X; step_ind++){
          /*printf("%f \n", daisies[step_ind][0]-15);*/
```

```
147         if ((int)daisies[step_ind][0]-15 >= 0) count_a_values[(int)daisies[step_ind][0]-15]++;
        }
149     for(countingA=0; countingA<70; countingA++){
          if(count_a_values[countingA]){
151             nr_of_A_values++;
          }
153       }

155     /*printf("%d \n", (int)daisies[step_ind][0]-15); */

157     if((alpha*R_sum+betha*(P-R))<P_inc){
          reg_time++;
159       if(reg_time==200){reg_time = 0;
          t_regulated++;}
161     }

163     fprintf(file, "%f, %f, %f, %f  %f, %d, %f, %f \n", P, R, meanA/(float)INDIVIDUALS_X, sum_theta, alpha*
            R_sum, nr_of_A_values, t_regulated, mew);
        meanA = 0;
165     sum_theta = 0;
        R_sum = 0;
167     nr_of_A_values = 0;
      }
169   /*fprintf(reg, "%f, %f \n", mew, t_regulated);*/
      /*t_regulated = 0;*/
171   t_regulated = 0;

173


175   fclose(file);
      fclose(reg);
177   Free2DdoubleArray(daisies, INDIVIDUALS_X);
      /*free(count_a_values);*/
179   return 0;
    }
181
    double** Make2DDoubleArray(int arraySizeX, int arraySizeY) {
183   double**theArray;
      int i;
185   theArray=(double**)malloc(arraySizeX*sizeof(double*));
      for(i=0;i<arraySizeX;i++)
187     theArray[i]=(double*)malloc(arraySizeY*sizeof(double));
      return theArray;
189 }

191 double random_number_inrange(int LOW, int HIGH){
      /*Variables to hold random values for the first and the second die on each roll.*/
193   double A_nr;     /*Declare variable to hold seconds on clock.*/
      A_nr = ((double) rand()/ MAX_RAND)* ((double)HIGH - (double)LOW) + (double)LOW;/*Get two new random
          values.*/
195   /*printf("%f \n", A_nr);*/
      return A_nr;
197   /* Modified from: http://www.cprogramming.com/tutorial/random.html */
    }
199


201


203 double fitness(double A, double R){
      double fitness = 0;
205   int count = 0;
      if ( R >= 15 && R <= 85 ){
207     if (fabs(A-R) <= pow(lambda, -0.5))fitness = 1-lambda*pow((A-R),2);
      }
209   return fitness;
    }
211
    int tournament(double **daisies, double R, int step_ind){
213   int daisy_min = 0;     /* Minimum daisies */
      int daisy_max = INDIVIDUALS_X;  /* Maximum daisies */
215   int rand_daisy1, rand_daisy2;    /* Random daisies */

217   /* Get random daisies */
      rand_daisy1 = (int)random_number_inrange(daisy_min, daisy_max);
219   rand_daisy2 = (int)random_number_inrange(daisy_min, daisy_max);
```

```
221   /* Calcelate fitness */
      if (fitness(daisies[rand_daisy1][0], R) > fitness( daisies[rand_daisy2][0], R)){
223     return rand_daisy1;
      }
225   else if (fitness(daisies[rand_daisy1][0], R) < fitness( daisies[rand_daisy2][0], R)){
        return rand_daisy2;
227   }
      else{
229     return step_ind;
      }
231 }

233 void Free2DdoubleArray( double **theArray, int arraySizeX ) {
    int i;
235 for (i=0;i<arraySizeX;i++){
      free(theArray[i]);
237 }
    free(theArray);
239 }
```

daisyworld18Nov.c