Optimus - Tutorial 3

Nicholas Andre G. Johnson and Aleksandr B. Sahakyan

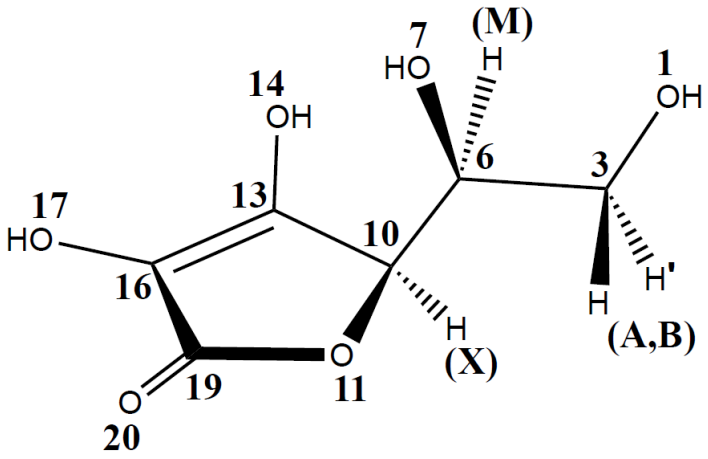
17 August 2018

Example 3: Vitamin C Molecule Geometry Optimisation

The focus of this Tutorial is to depart from problem classes involving the search for functions to represent data and demonstrate how Optimus can be flexibly applied to arbitrary problem classes provided that they are formulated in accordance with Optimus specifications. Additionally, this Tutorial will illustrate that Optimus can act as an optimisation kernel while calling external programs to execute a significant amount of the necessary computation for the optimisation process.

In this Tutorial, Optimus will be used to solve a molecular geometry optimisation problem. Specifically, Optimus will be used to determine the values of two dihedral angles in the L-ascorbic acid (Vitamin C) molecule such that the molecule is in its ground state energy configuration. Vitamin C was selected to be the studied molecule because it has more than one freely rotating carbon-carbon sigma bond and the potential for intramolecular hydrogen bonding due to the presence of multiple hydroxy groups and lone pair donating oxygen atoms. Moreover, Vitamin C is not a particularly large molecule. Due to these circumstances, Vitamin C has a non trivial groud energy state (as opposed to a molecule like ethane for instance) but one that does not require several days or weeks of simulation to arrive at (the optimisation procedures below took roughly 14-18 hours to terminate).

This is the molecular structure of Vitamin C:

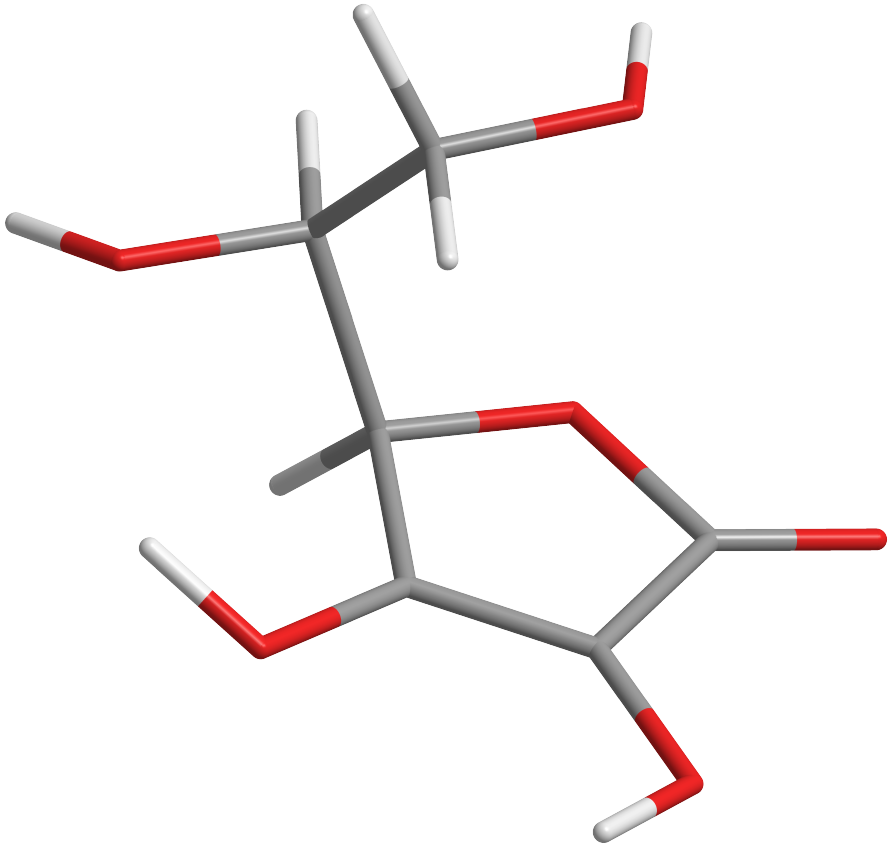


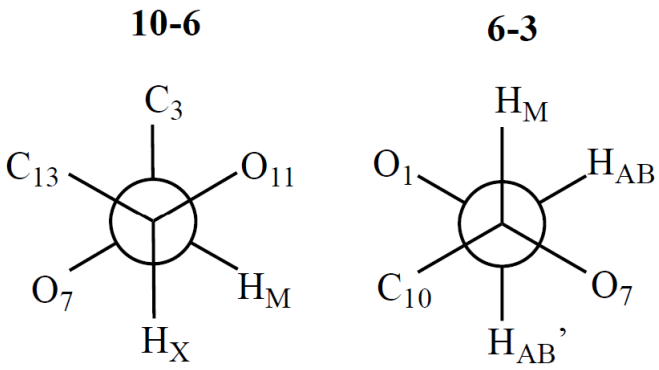
Free rotation is possible around two sigma bonds in this structure: the bond joining carbon 3 and 6, and the bond joining carbon 6 and 10. The ground state configuration of Vitamin C will be a configuration such that steric clash between the constituent members bonded to the freely rotating bonds is minimized while also allowing for close proximity between hydrogen bond donating and hydrogen bond accepting groups. In the following sections, we formalize this optimisation problem and use Optimus to arrive at the solution.

Defining Optimus Inputs

As in the previous Tutorials, we must first rigorously define the parameters which we are optimising. Let us begin by defining a dihedral angle as it applies to molecular geometry: a dihedral angle is the angle between two intersecting planes, where each plane is specified by 3 atoms of which 2 are common between both planes.

Thus, a total of 4 atoms are needed to specify a dihedral angle. The conformation of Vitamin C with respect to its two freely rotating sigma bonds can be specified via two dihedral angles. Let ψ be the dihedral angle defined by the atoms numbered 1, 3, 6 and 7 and let ϕ be the dihedral angle defined by the atoms numbered 6, 7, 10 and 11. Loosely, ψ can be thought of as the angle between oxygen atoms 1 and 7 while ϕ can be thought of as the angle between oxygen atoms 7 and 11. Having defined these two angles, we can now define the parameter set K as a numeric vector of length 2 whose entries are ψ and ϕ. We will arbitrarily initialize ψ and ϕ to have value 180. The corresponding Vitamin C conformation is illustrated below using a 3D structure and Newman Projections along the two rotatable carbon-carbon bonds:





In the 3D structure, grey denoted Carbon, red denotes Oxygen and white denoted Hydrogen.

```
K <- c(PHI=180, PSI=180)
```

Now we will specify a model function $m()$ which will operate on K. Starting from an arbitrary molecular conformation, altering the value of K will likely cause certain clashes or non-optimal interactions between atoms in the molecule that are not used in the definition of the angles ψ and ϕ. As such, after receiving an input set of parameters K, $m()$ will have to alter the 3D location of constituents atoms while holding K fixed to arrive at the most stable geometry for the input K. Here, unlike in previous Tutorials, to accomplish this task $m()$ will call a completely external program: MOPAC (2016). MOPAC is a geometry optimiser that uses EigenFollowing to arrive at a local minimum (note that calling MOPAC for a single instance does not guarantee a global minimum will be found). MOPAC takes as input the specification of an initial molecular geometry in addition to an indication of which molecules the program is able to displace (or angles it can alter) and outputs a nearby local minimum molecular configuration with its corresponding energy in kcal/mol. For this optimisation problem, the input to MOPAC will be structured as a Z matrix, a common form for describing a molecular configuration which consists of using lengths, angles and dihedral angles with respect to previously defined atoms to define new atoms in the configuration.

The function $m()$ will construct a Z matrix for Vitamin C using the input dihedral angles K and default values for the remaining relationships needed to define the molecule. $m()$ will then call MOPAC with the newly constrcuted Z matrix, specifiying that all relationships may be altered except the input dihedral angles K. Finally, $m()$ will return the energy calculated by MOPAC.

Note that to avoid non convergence issues when calling MOPAC, $m()$ returns a default energy value of -100 kcal/mol if a call to MOPAC does not terminate within 10 seconds. Also, note that although $m()$ requires no additonal data on top of K to operate, $m()$ must still be defined to take an input DATA in accordance with Optimus specifications. Lastly, note that a local installation of MOPAC (2016) is required to execute this optimisation procedure. Below is the definition of $m()$:

```
m <- function(K, DATA = NULL){

  notconvergedE = -100.00
  # this should be your local path to MOPAC
  mopac.cmd="/home/group/prog/mopac2016/MOPAC2016.exe"
  clean = TRUE

  # MOPAC semiempirical QM input file preparation, with given PHI and PSI
  # dihedral angles.

  geo <- c(
    "RHF PM6 EF GEO-OK MMOK T=10 THREADS=1",
    "Vitamin C with two controllable dihedral angles psi(7,6,3,1) and phi(11,10,6,7)",
    "  ",
    "O     0.00000000  0    0.0000000  0    0.0000000  0     0     0     0",
    "H     0.98468620  1    0.0000000  0    0.0000000  0     1     0     0",
    "C     1.43651250  1  110.7230618  1    0.0000000  0     1     2     0",
    "H     1.10751723  1  103.6603154  1 -167.5282722  1     3     1     2",
    "H     1.10658657  1  110.2236860  1  -51.3620456  1     3     1     2",
    "C     1.53950336  1  112.8074046  1 -123.2791585  1     3     4     5",
    paste0("O     1.42824262  1  103.4315186  1 ",K["PSI"]," 0     6     3     1"),
    "H     0.99584949  1  109.9022382  1 -165.7055126  1     7     6     3",
    "H     1.11472171  1  108.4417082  1   75.1535637  1     6     7     8",
    "C     1.54244170  1  109.4042184  1 -120.8240216  1     6     7     9",
    paste0("O     1.46313669  1  105.7792445  1 ",K["PHI"]," 0    10     6     7"),
    "H     1.11252563  1  112.8336666  1 -114.5813834  1    10     6    11",
    "C     1.51686608  1  113.4849244  1 -112.8332453  1    10    12    11",
    "O     1.34410484  1  125.3617342  1  179.6090511  1    13    10    11",
    "H     1.03381724  1  110.9736522  1  -13.3419919  1    14    13    10",
    "C     1.36084908  1  124.8906459  1  167.6242325  1    13    14    15",
```

```
    "O     1.35614887  1  131.9374989  1   -0.0333000  1    16    13    14",
    "H     1.00338885  1  109.4220239  1    0.3798200  1    17    16    13",
    "C     1.49109250  1  118.0837177  1 -179.7749947  1    16    17    18",
    "O     1.18961787  1  136.9144035  1   -0.6060924  1    19    16    17",
    "  "
  )

  # Submitting the MOPAC optimisation job, where all the spatial parameters
  # are relaxed except the pre-set PHI and PSI angles. The job is run requesting
  # maximum 10 seconds of time limitation. Most (if not all) complete within
  # half a second.
  random.id <- as.character(sample(size=1, x=1:10000000))
  write(geo, file=paste0(random.id,".mop"))
  system(paste0(mopac.cmd," ",random.id,".mop"))

  if( file.exists(paste0(random.id,".arc")) ){
    e.line <- grep("HEAT OF FORMATION",
                   readLines(paste0(random.id,".arc")),
                   value=TRUE)
    e.line <-  strsplit(e.line," ")[[1]]
    O <- as.numeric(e.line[e.line!=""][5])
  } else {
    O <- notconvergedE
  }

  if(clean){
    file.remove(grep(random.id, dir(), value=TRUE))
  }

  return(O) # heat of formation in kcal/mol
}
```

Next, we define the function $u()$ with returns an energy E and a quality Q of the candidate solution. Since the $m()$ will already output a value for the physcial energy of the candidate Vitamin C configuration, $u()$ can simply set E to be the return value of $m()$. We will make $u()$ set Q to be the negative of the return value of $m()$ such that candidate configurations with lower energies produce higher values of Q. Again, although $u()$ does not require any additional data to accomplish this functionality, it must nevertheless be written to optionally accept an input parameter DATA.

```
u <- function(O, DATA = NULL) {
    result <- NULL
    result$Q <- -O
    result$E <- O
    return(result)
}
```

Finally, we define the function $r()$. $r()$ will randomly select either ψ or ϕ to alter. Thereafter, $r()$ randomly increases or decreases the selected angle by 2 degrees. $r()$ will also ensure that $\psi, \phi \in [-180.0, 180.0]$ throughout the optimisation process.

```
r <- function(K) {
    K.new <- K
    # Setting the alteration angle to 3 degrees:
    alter.by <- 2
    # Randomly selecting a term:
```

```
    K.ind.toalter <- sample(size = 1, x = 1:length(K.new))
    # Altering that term by either +alter.by or -alter.by
    K.new[K.ind.toalter] <- K.new[K.ind.toalter] + sample(size = 1, x = c(alter.by,
        -alter.by))

    # Setting the dihedral angles to be always within the -180 to 180 range.
    if (K.new[K.ind.toalter] > 180) {
        K.new[K.ind.toalter] <- K.new[K.ind.toalter] - 360
    }

    if (K.new[K.ind.toalter] < -180) {
        K.new[K.ind.toalter] <- K.new[K.ind.toalter] + 360
    }

    return(K.new)
}
```

The process of determining the energy of a configuration corresponding to a given set of angles ψ, ϕ is the most computationally intensive part of this optimisation formulation. Having defined the necessary inputs for Optimus, it should be apparent that this calculation will entirely be handled by MOPAC. This ability to serve as an optimisation kernel by off-loading a significant amount of computation to an external program is one of the many strengths of Optimus.

Defining a Benchmark Solution

Before calling Optimus, we must establish a benchmark solution which will be used to evaluate the efficacy of Optimus. In order to explore the energy landscape associated with the parameter space of ψ and ϕ, a PM6 optimization was performed on 10 conformers from an MM2 optimization run (the details of PM6 and MM2 are not important for the purposes of this tutorial) which resulted in the identification of 7 local energy minima, shown in the table below (listed in increasing order by energy):

Table 1: 7 Vitamin C Conformational Local Minima

	E (kcal/mol)	PHI	PSI
Conformation 1	-233.206	92.58	74.19
Conformation 2	-232.877	50.60	-172.79
Conformation 3	-231.800	-169.67	-41.23
Conformation 4	-230.822	47.43	-166.61
Conformation 5	-230.274	-172.20	-54.45
Conformation 6	-225.214	-75.69	-104.44
Conformation 7	-224.875	-73.31	155.02

We will assume that the above listed conformations represent all conformational local minima in the parameter space of ψ and ϕ. Under this assumption, Conformation 1 should be considered the ground state conformation of Vitamin C. The accuracy of the results produced by Optimus can thus be judged by comparing them to the data listed in this table. It is important to recognize that the "resolution" of ψ and ϕ when being optimized by Optimus is 2 degrees due to the manner in which $r()$ was defined. As such, results produced by Optimus that are within plus or minus 2 degrees of a reference conformation should be tolerated.

Acceptance Ratio Annealing Optimus Run

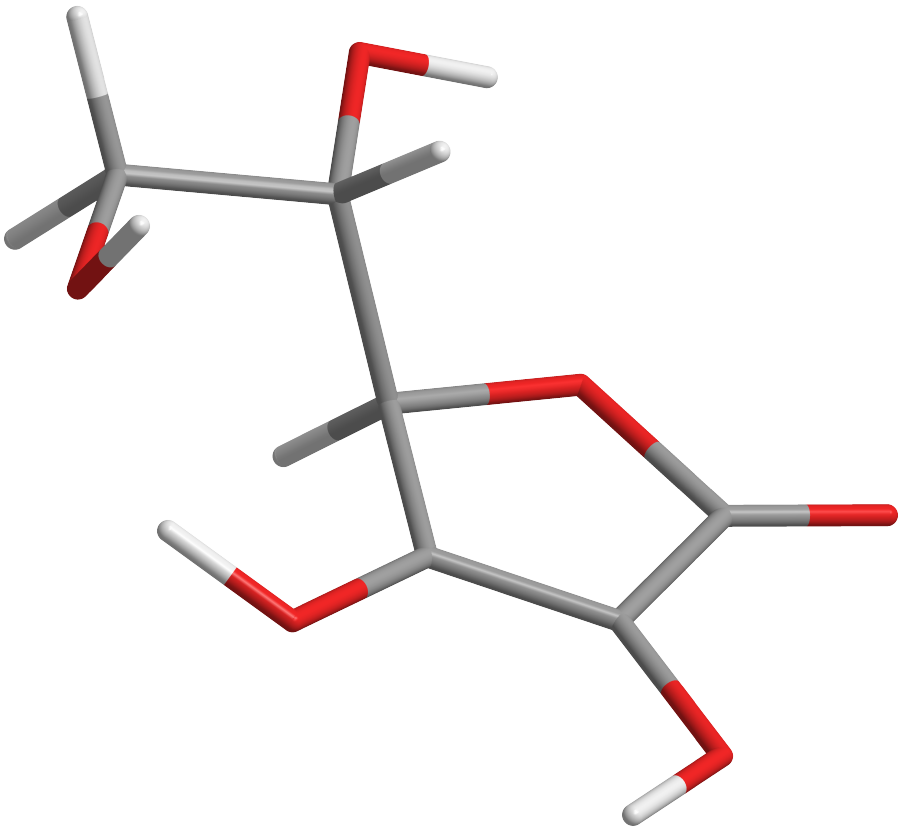
For the Acceptance Ratio Annealing run, we will set NUMITER = 100 000 because each optimsation step is more costly due to the relatively computationally expensive calls to MOPAC. Moreover, we will set CYCLES = 2. Although this shortens the length of an annealing cycle to 50 000 steps (whereas 100 000 steps per cycle has been kept constant over the previous tutorials), having more than 1 annealing cycle is likely more beneficial than insisting on a cycle lasting 100 000 steps as opposed to 50 000.

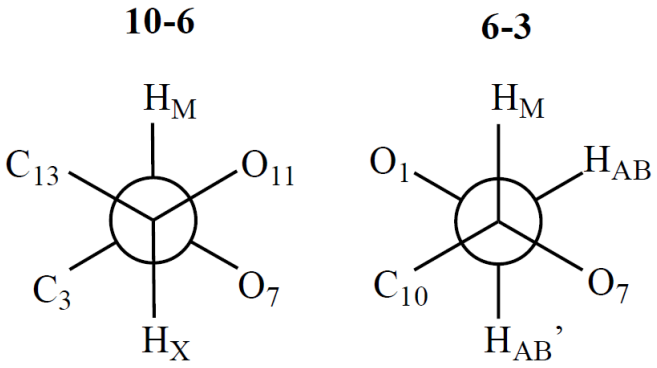
```
Optimus(NCPU = 4, K.INITIAL = K, rDEF = r, mDEF = m, uDEF = u, OPT.TYPE = "SA",
    OPTNAME = "vitamin_4_SA", NUMITER = 1e+05, CYCLES = 2, DUMP.FREQ = 50000,
    LONG = FALSE)
```

Table 2: 4 Core Acceptance Ratio Annealing Optimus Run Results

	E (kcal/mol)	PHI	PSI
Processor 1	-232.874	50	-172
Processor 2	-232.353	-158	30
Processor 3	-232.874	50	-172
Processor 4	-232.874	50	-172

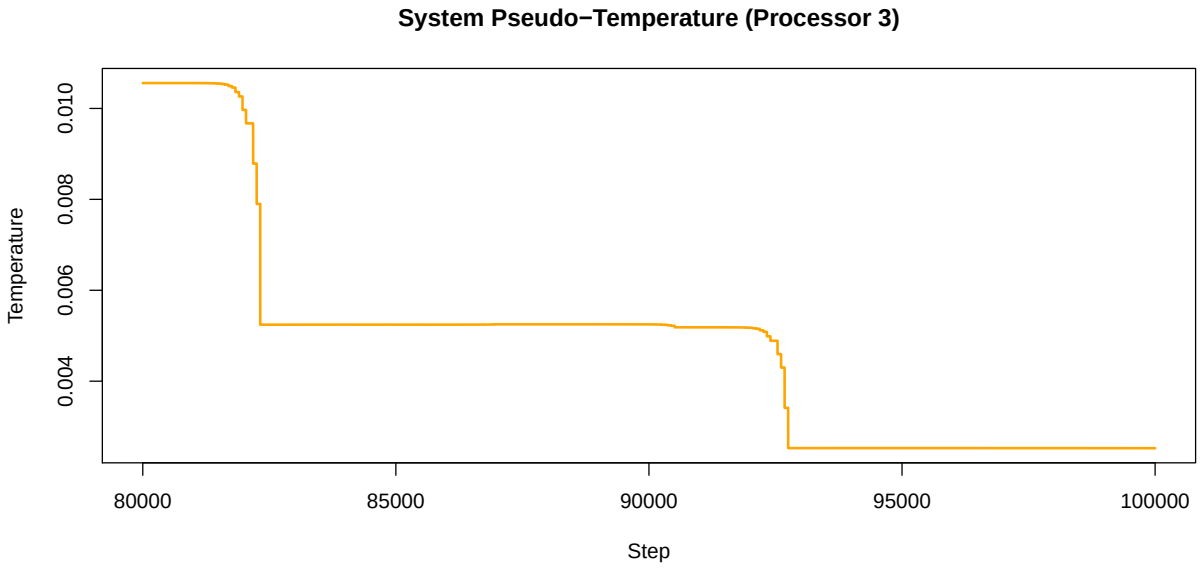
Processors 1, 3 and 4 all arrived at a conformation defined by $\{\phi = 50, \psi = -172\}$, with an energy of -232.874 kcal/mol. The below 3D structure and Newman Projections depict this solution:

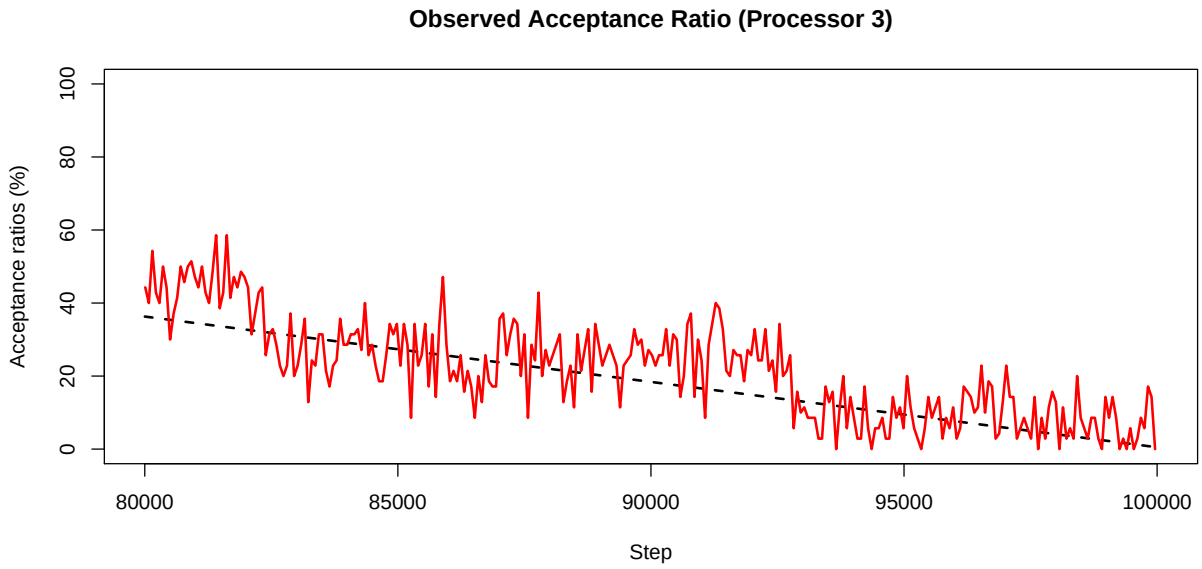




This conformation is equivalent to benchmark Conformation 2. Thus, in this example, Acceptance Ratio Annealing was able to find the Vitamin C conformation with the second lowest energy in the parameter space. This performance is strong, especially given that the energy difference between Conformation 1 and Conformation 2 is only -0.329 kcal/mol.

The graphs below illustrate the system psuedo-temperature and observed acceptance ratio for the last 20 000 optimisation iterations executed by Processor 3.





Replica Exchange Optimus Run

Let us now consider the Replica Exchange version of Optimus on 12 processors with the variable ACCRATIO defined as in the previous Tutorials.

```
ACCRATIO <- c(90, 82, 74, 66, 58, 50, 42, 34, 26, 18, 10, 2)
```

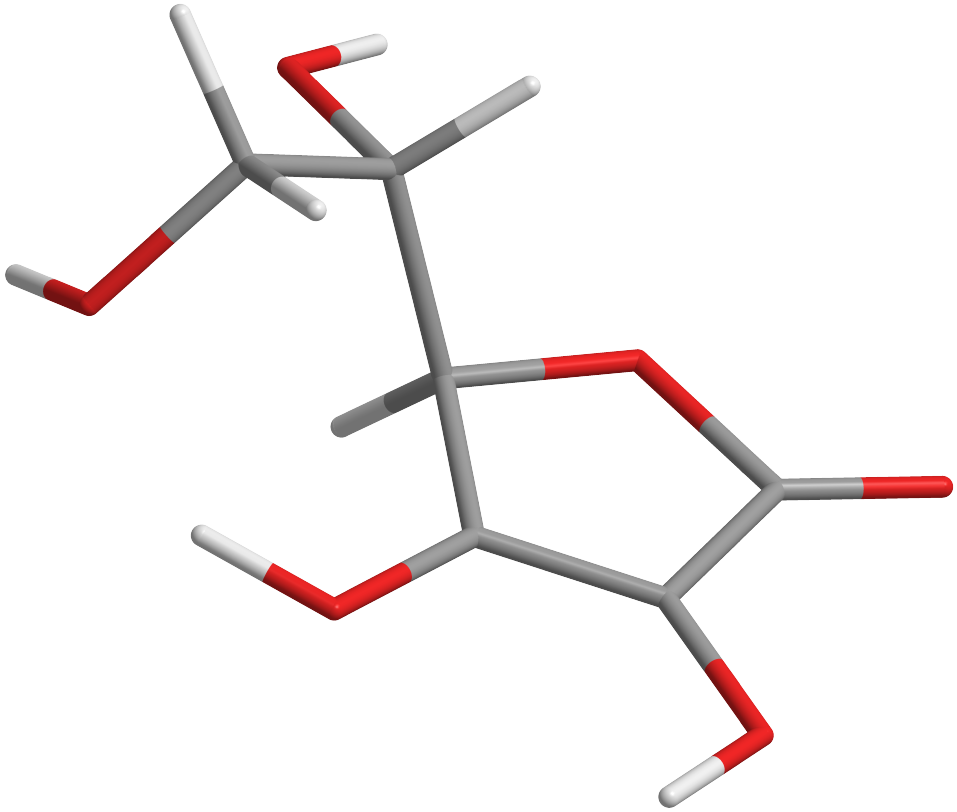
Just as in the Acceptance Ratio Annealing run, we will set NUMITER = 100 000. Moreover, we will set EXCHANGE.FREQ = 500 such that the number of iterations between subsequent exchanges between replicas is 200 as it was in Tutorial 2. For the same reasons as in Tutorial 2, we will set STATWINDOW = 50 for the Replica Exchange run.

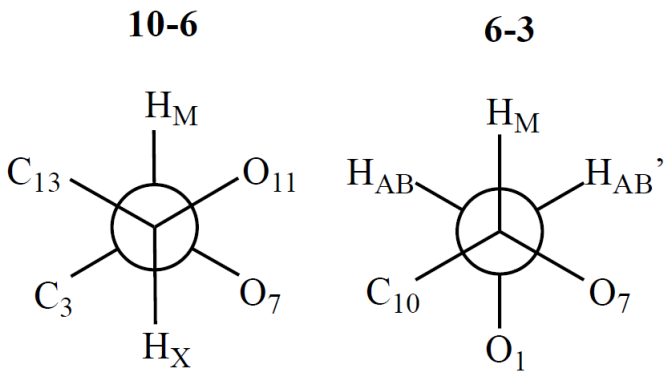
```
Optimus(NCPU = 12, K.INITIAL = K, rDEF = r, mDEF = m, uDEF = u, ACCRATIO = ACCRATIO,
    OPT.TYPE = "RE", DATA = DATA, OPTNAME = "vitamin_12_RE", NUMITER = 1e+05,
    EXCHANGE.FREQ = 500, STATWINDOW = 50, DUMP.FREQ = 50000, LONG = FALSE)
```

Table 3: 12 Core Replica Exchange Optimus Run Results

	Replica Acceptance Ratio	E (kcal/mol)	PHI	PSI
Processor 1	90	-229.2359	-164	-178
Processor 2	82	-229.2359	-164	-178
Processor 3	74	-233.1453	82	84
Processor 4	66	-233.1979	90	76
Processor 5	58	-229.2359	-164	-178
Processor 6	50	-232.8742	50	-172
Processor 7	42	-233.1947	94	74
Processor 8	34	-229.2359	-164	-178
Processor 9	26	-229.2359	-164	-178
Processor 10	18	-227.6394	180	158
Processor 11	10	-229.2359	-164	-178
Processor 12	2	-229.2359	-164	-178

Of the 12 replicas, Processor 4 recovered the conformation with the lowest energy (−233.1979), defined by $\{\phi = 90, \psi = 76\}$. The below 3D structure and Newman Projections depict this solution:



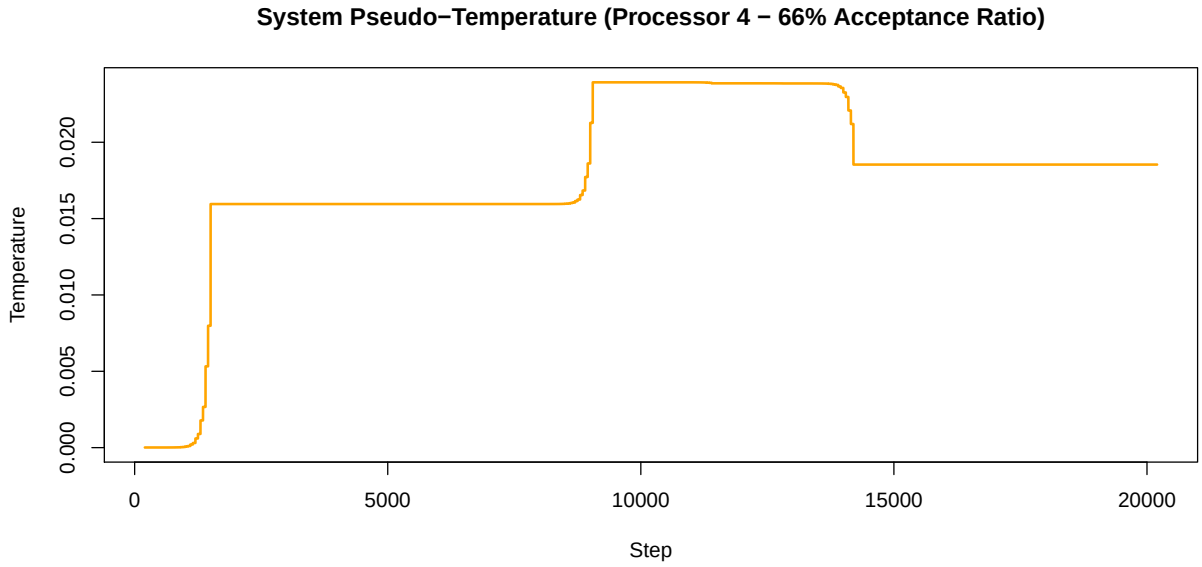


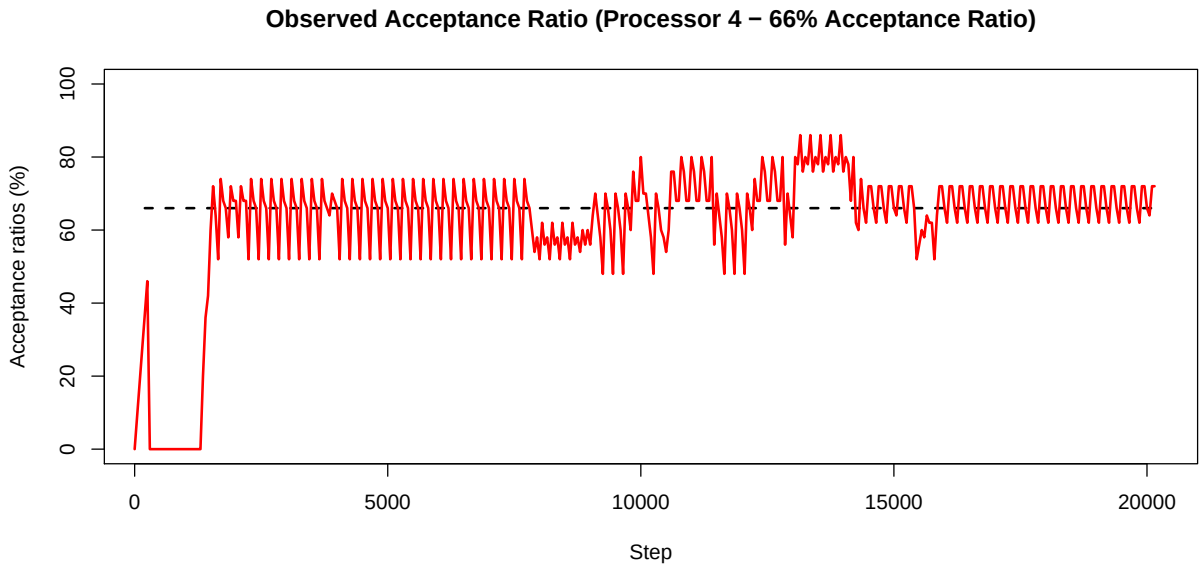
This solution corresponds to reference Conformation 1, the global minimum energy state for Vitamin C. Thus, for this optimisation problem, the Replica Exchange version of Optimus outperformed Acceptance Ratio Annealing by succeeding in finding the global minimum of the energy landscape while the latter version found only the second lowest local energy minimum.

If we compare the solution found by Processor 4 to benchmark Conformation 1, it is evident that the value for ϕ found by Optimus lies slightly outside of the plus or minus 2 degree window that was discussed earlier. Contrarily, Processor 7 finds a solution $\{\phi = 94, \psi = 74\}$ which does lie strictly within the resolution window. Despite this, the solution of Processor 4 has a slightly lower energy (−233.1979) than the solution of Processor 7 (−233.1947) and so represents a better solution. Given that the local energy landscape around Conformation 1 is unknown, it is not unreasonable that within this local area, a conformation "closer" to Conformation 1

does not necessarily result in a lower energy. Finally, notice that Replica 6 recovered the same conformation that was identified by Acceptance Ratio Annealing Optimus.

The below graphs illustrate the system psuedo-temperature and observed acceptance ratio for the first 20 000 optimisation iterations executed by Processor 4 (66% acceptance ratio replica).





When the optimisation process is first initialized, it is very unlikely that the input initial temperature is conducive to the target acceptance ratio. As such, the Temperature Control Unit alters the system pseudo-temperature considerably and rapidly to align the observed acceptance ratio with the target acceptance ratio, as can be seen in the above two graphs. Moreover, as stated in the previous Tutorial, an exchange between two replicas often has a similar effect of introducing a parameter configuration that is not conducive to the current system pseudo-temperature, which catalyzes significant temperature adjustments executed by the TCU. Accordingly, sharp increases or decreases in the system pseudo-temperature and significant changes in the value around which the observed acceptance ratio oscillates in the graph above likely indicate steps at which an exchange involving replica 4 occurred.

Summary

We are now familiar with how to structure a molecular geometry optimisation problem to be solved with Optimus as a kernel while interfacing with an entirely external program, MOPAC, to handle a significant amount of the necessary computation. Using 7 local minima identified by a PM6 optimization of 10 conformers from an MM2 run as validation, we saw that the Acceptance Ratio Annealing mode of Optimus was able to find the second lowest local minima while the Replica Exchange mode recovered the global energy minimum (outperforming Acceptance Ratio Annealing).

Table 4: Summary of Solutions

	Energy (kcal/mol)	PHI	PSI
Ground State Reference	-233.2060	92.58	74.19
Optimus (Acceptance Ratio Annealing)	-232.8740	50.00	-172.00
Optimus (Replica Exchange)	-233.1979	90.00	76.00