# Web Visualization of a Trajectory Generated from the General Mission Analysis Tool (Part 3)

By Daniel A. O'Neil

## Introduction

A Web-based Mission Visualization System (WMVS) needs a standard file format for storing orbital trajectories. Part 3 of this tutorial series explains how to write a JavaScript web-app to convert a fixed width text data file exported from the General Mission Analysis Tool (GMAT) and convert the file into JavaScript Object Notation (JSON). Assuming the reader has read the previous tutorials, he or she may recall that Part 1 explained how to perform this conversion in Excel. In addition to the converter, this tutorial will explain how to create a generic web-based mission visualizer that reads JSON files.

## Prerequisites and Objectives

The trajectory file converter and the generic web-based mission visualizer rely upon the HTML5 File Reader Application Programming Interface (API). Several websites and books provide tutorial on how to use the File Reader API. Chuck Hudson and Tom Leadbetter's book "HTML5 Developer's Cookbook" provided example code that enabled the development of the file converter described in this tutorial. Figure 1 presents the fixed width text file containing Cartesian coordinates and the user interface of the JSON converter. A Browse button on the converter enables selection of a text file.

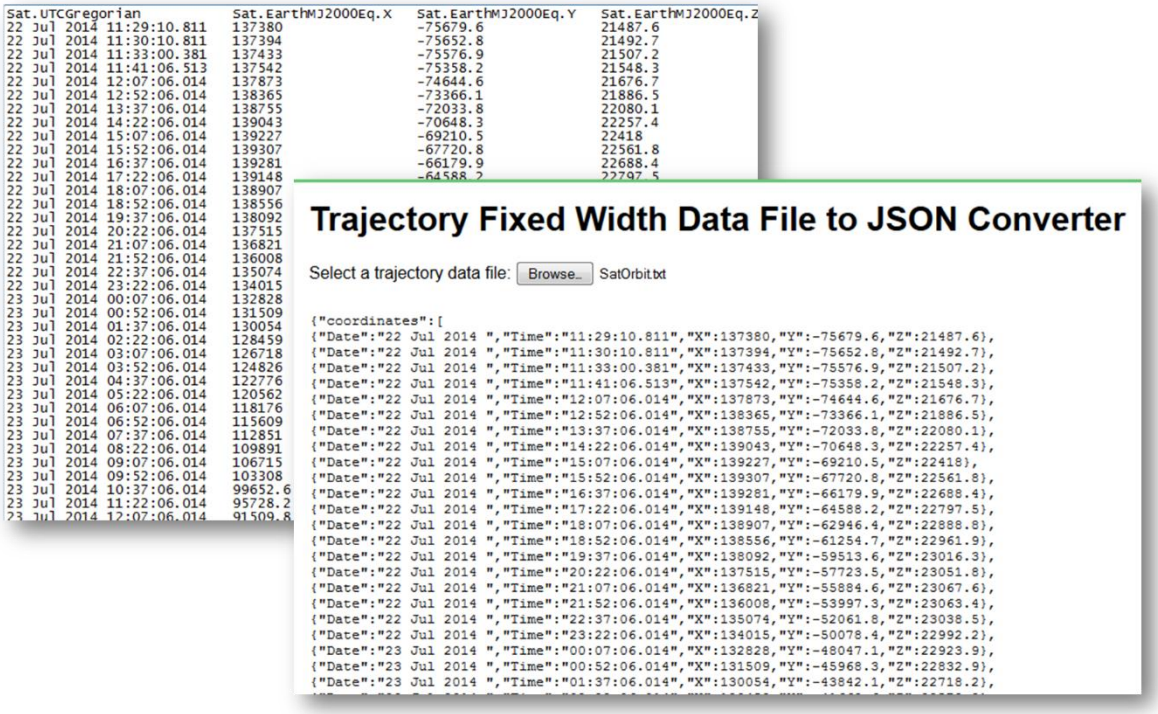


*Figure 1 A text file with Cartesian coordinates and an a generated JSON file*

Objectives of this tutorial include:

1. Present a walk-through of the code within the text to JSON converter
2. Explain how to incorporate the File Reader API functions into a visualization
3. Provide the source code in the appendices

## Fixed Width Text to JavaScript Object Notation Conversion Demo

Part 1 of this tutorial series explained how to generate a fixed width text file from the GMAT. Such a file can provide date, time, and Cartesian coordinates for a spacecraft. This section of the tutorial provides a code walk-through of a JavaScript web-app that converts a trajectory text file into JavaScript Object Notation (JSON). A subsequent section will walk-through a web-based mission visualization that uses the JSON object. Appendix A provides the source code for the text to JSON converter. Appendix B provides the source code for the web-based mission visualization that uses the JSON object. Appendix C provides an example trajectory data file that was exported from the GMAT. The following Uniform Resource Locator (URL) leads to the text to JSON converter, which was depicted in Figure 1.
http://daoneil.github.io/spacemission/X3Dom/Data_to_JSON_Converter6.html

The reader may use a trajectory data file produced during Part 1 of the tutorial or use the trajectory data provided in Appendix C. If you are using the data from Appendix C, please copy and paste the data into a text file named MolniyaOrbit.txt. After opening the text to JSON converter in your web-browser, click on the Browse button to select a local file on your computer. Navigate to the appropriate directory or folder and select MolniyaOrbit.txt. Within a second or two, the web-page display the trajectory in JSON format as depicted in Figure 1. Copy and paste the JSON formatted data into a text file and name it MolniyaOrbit.json.

## Web-Based Mission Visualization Using an Uploaded JSON File Demo

Figure 2 presents a screenshot of a web-based mission visualization that uses an uploaded JSON file. The following URL leads to this demonstration.

http://daoneil.github.io/spacemission/X3Dom/JSON_Trajectory_Reader_with_Model_appendix.html

A Browse button enables you to navigate through your directories or folders to select a local file. Please select the MolniyaOrbit.json file that you generated in the previous section. After uploading the JSON file, the simulation will display a satellite moving along the

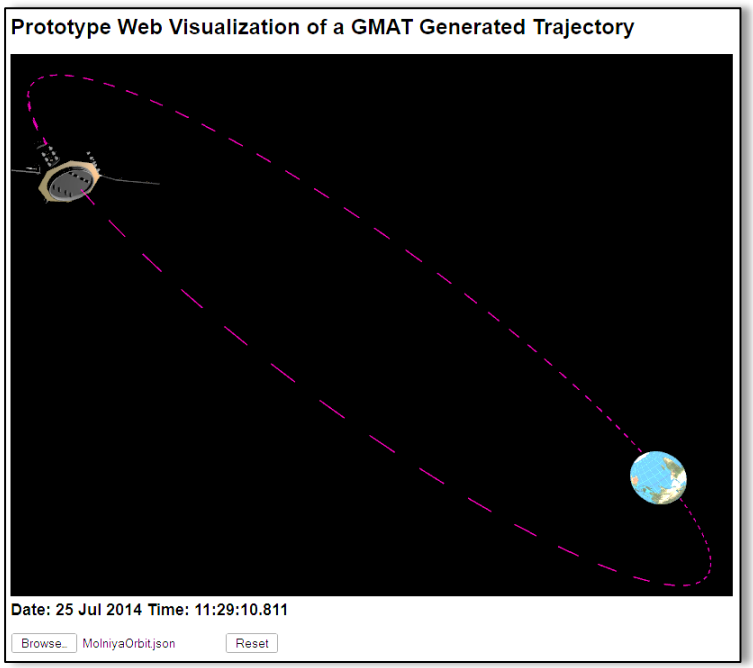


*Figure 2 Web-based mission visualization that displays an uploaded JSON object*

trajectory. With your mouse, you can interact with the simulation:

- Right-button click and drag to rotate the model
- Left-button click and drag to zoom in and out
- Middle-button click and drag to pan around the scene

When the simulation stops, you can restart it by clicking the Reset button.

## Text to JSON Converter Code Walk-Through

This section presents a few code snippets. Refer to Appendix A to view the code snippets within the context of the program. Within the header, a Cascading Style Sheet (CSS) section formats the text area:

```
<style>
…
#fileDisplayArea {
  margin-top: 2em;
  width: 100%;
  overflow-x: auto;
}
</style>
```

After loading, the page calls the handleFile function, i.e., `<body onload="handleFile()">`.

The handleFile function adds an event listener to a file input button. If there is a change to the file input, a file variable gets the file name.

```
function handleFile() {
    var fileInput = document.getElementById('fileInput');
    …
    fileInput.addEventListener('change', function(e) {
          var file = fileInput.files[0];
```

The HTML5 File Reader API includes several events that you overwrite. The following code snippet overwrites the onload event. Comments explain the first few lines. The last line is a for loop that steps through each line of the text file.

```
reader.onload = function(e) {
            dat = reader.result;        // Get the contents of the file.
      ...
      while (curChar != "\n") {           // Find the end of line.
            lineWidth = lineWidth + 1 ;  // Count characters in a line.
            curChar = dat[lineWidth] ;
         }
            curPos = lineWidth + 1;       // Advance to the next line.

            var iNumLines = dat.length/lineWidth ;
        ...
      for (var i = 0; i < iNumLines; i++) {
      ...
```

Within the for loop there are several calls to an extract data function, which parses a slice of the current line of text and returns the text string for a record variable named rec. The following code snippet excludes the calls to extract data, so refer to the appendix. The following codes

snippet presents an if branch and determines whether there are more lines in the text file. After populating the rec variable with the date, time, and Cartesian coordinates, the fileDisplayArea InnerHTML property is populated with the JSON formatted data. The closing bracket with the comment about the next line of text closes the for-loop in the previous code snippet. The next few lines change the comma to square and curly brackets to complete the JSON object. A call to the readAsText event causes text reader to load the text file. If the file is not of type text, an error message will be written into the file display area.

```
if (curPos < totalChar) {       // Have we reached end of the file?
        ...
            fileDisplayArea.innerHTML = fileDisplayArea.innerHTML +
                          "{" + '"Date":' + '"' + rec.date + '"' + ","
                              + '"Time":' + '"' + rec.time + '"' + ","
                                          + '"X":' + rec.X + ","
                                          + '"Y":' + rec.Y +  ","
                                          + '"Z":' + rec.Z  ;
            fileDisplayArea.innerHTML = fileDisplayArea.innerHTML  + "}, \n"
} // end if
        ...
   } // next line of text
            var sTraJSON = fileDisplayArea.innerHTML.toString() ;
            var iLastCharNum = sTraJSON.length ;

            sTraJSON = sTraJSON.substring(0,iLastCharNum -3) ;
            sTraJSON = sTraJSON + "]}" ;

            fileDisplayArea.innerHTML = sTraJSON ;
        }
             reader.readAsText(file) ;

        } else {
            fileDisplayArea.innerText = "File not supported!"
                  }
        });
```

## Web-Based Mission Visualization using a JSON File Code Walk-Through

Both Part 1 and Part 2 of this tutorial series presented X3D code. Part 2 explained how to texture-map a globe. Appendix B includes the same X3D code to display the Earth, so this section will not walk-through that code; refer to Part 2 for a walk-through of the X3D code.

When the web-based mission visualization web-page loads, it calls the startUpdate function, i.e., `<body onload="startUpdate()">`. This function reads the JSON file, determines the initial position, reads an X3D file for the satellite, and then steps through each line in the JSON file to draw the trajectory and move the satellite model.

The following code snippet includes the declaration of the startUpdate function, which includes a function to retrieve a selected file and the File Reader API onload event function to return the result from the file reader. The mission variable contains the parsed JSON file that provides the date, time, and Cartesian coordinates that define the trajectory.

```
function startUpdate() {
...
 function handleFileSelect(evt) {
    var files = evt.target.files; // FileList object
     f = files[0];
      var reader = new FileReader();

      // Closure to capture the file information.
      reader.onload = (function(theFile) {
        return function(e) {

      mission = JSON.parse(e.target.result);   // Parse trajectory object.
```

A for-loop within the onload function iteratively stores the mission coordinates in a segment coordinates variable named segCoords and appends the segment coordinates to an orbitCoords string. An X3D line object uses the orbitCoords string to position points that draw the orbit. These actions are expressed in the following code snippet.

```
for (var segment in mission.coordinates) {
  …
var segCoords = [mission.coordinates[segment].X/10000, …

orbitCoords = orbitCoords + segCoords[0]  + " " + segCoords[1] …

var line = document.createElement('IndexedLineSet');
    line.setAttribute("coordIndex", segIndex);
var coords = document.createElement('Coordinate');
    coords.setAttribute("point", orbitCoords);
```

Part 2 of this tutorial series generated a green block to represent a satellite. The web-app in Appendix B goes a step further by including an X3D model of a satellite. The first five lines of the following code snippet establishies an initial position and creates a transform with attributes to translate the model to the position and scale it An Inline command links the X3D model of a satellite. The satellite model is appended as a child of the Earth. Finally, a setInterval command periodically calls an update position function.

```
/*-----------------------------
 *  Create a satellite model
 *-----------------------------*/
 var pos = [mission.coordinates[0].X/10000, mission.coordinates[0].Y/10000,
            mission.coordinates[0].Z/10000]
  var t = document.createElement('Transform');
      t.setAttribute("translation", pos[0] + " " + pos[1] + " " + pos[2] );
      t.setAttribute("id", 'satPosition');
      t.setAttribute("scale", 0.07 + " " + 0.07 + " " + 0.07) ;

      var satModel = document.createElement('Inline') ;
      satModel.setAttribute('url', "TrianaSatellite.x3d") ;

      t.appendChild(satModel) ;
      var objsat = document.getElementById('theEarth');
      objsat.appendChild(t);
      setInterval(function () {updatePosition() }, 50);
        };
      })(f);
```

The last function in the web-app is the reset function, which sets the position of the satellite to the first position in the orbital trajectory.

## Future Work

The HTML5 File Reader API also provides a drag-and-drop capability. Potentially, a future version could add the drag-and-drop functions to enable uploading of multiple trajectory files. Other improvements could a user interface that enables selection from several spacecraft models. Additional user interface widgets could include a slider-bar for stepping through the orbital trajectory so the user can see the spacecraft at different points in the trajectory.

## Conclusion

This three part tutorial explained how to generate a trajectory data file from the GMAT, convert the file into JSON, and how to move an X3D model along the trajectory. The code walk-throughs and reusable example code ought to provide the reader a starting point for designing and developing a web-based mission visualization system.

## Appendix A -- Source code for the text to JSON converter

```
<!DOCTYPE html>
<html>
<head>
<title>Trajectory Data File Reader</title>
</head>
<style>
html {
  font-family: Helvetica, Arial, sans-serif;
  font-size: 100%;
  background: #333;
}

#page-wrapper {
  width: 850px;
  background: #FFF;
  padding: 1em;
  margin: 1em auto;
  min-height: 300px;
  border-top: 5px solid #69c773;
  box-shadow: 0 2px 10px rgba(0,0,0,0.8);
}

h1 {
      margin-top: 0;
}

img {
  max-width: 100%;
}

#fileDisplayArea {
  margin-top: 2em;
  width: 100%;
  overflow-x: auto;
}
</style>
<body onload="handleFile()">
<script>
var dat = new Blob();      // a blob to hold the content of the uploaded file
var rec = {} ;          // a record for date, time, and Cartesian coordinates
var records = new Array() ;  // an array of records  <-- Presently, not used.
var recCount = 0 ;           // a counter for the records in the array
var curPos = 0 ;          // current position within the current line of text
var curChar = "" ;       // current character within the current line of text
var curEOL = 0 ;         // calculated End Of Line using curPos + linewidth
var lineWidth = 0;            // number of characters in the line of text
var totalChar = 0 ;          // Calculated number of characters in the file

function extractData(someText) {
    var theData = "" ;
      var currentPosition = 0 ;
      var maxChar = someText.length ;

   while (theData == "") {
    if (someText[currentPosition] == " ") {
```

```
      // Find the next non-blank character.
         while (someText[currentPosition] == " ") {
                    currentPosition = currentPosition + 1 ;
            } // end while
      }   // end if
    else {
      // Update the global current position.
         curPos = curPos + currentPosition ;
         while (someText[currentPosition] != " ") {
               theData = theData + someText[currentPosition] ;
                   currentPosition = currentPosition + 1 ;
            } // end while
        } // end else
    } // end while

      return theData ;
} // end extractData

  function handleFile() {
      var fileInput = document.getElementById('fileInput');
      var fileDisplayArea = document.getElementById('fileDisplayArea');

      fileInput.addEventListener('change', function(e) {
            var file = fileInput.files[0];
            totalChar = file.size;       // Get total number of characters.
            var textType = /text.*/;
        if (file.type.match(textType)) {
            var reader = new FileReader();

            reader.onload = function(e) {
            dat = reader.result;    // Get the contents of the file.
            curChar = dat[0] ;      // Start at the first character
            while (curChar != "\n") {        // Find the end of line.
          lineWidth = lineWidth + 1 ;  // Count characters in line.
            curChar = dat[lineWidth] ;
      }
            curPos = lineWidth + 1;   // Advance to the next line.

            var iNumLines = dat.length/lineWidth ;
            fileDisplayArea.innerHTML = '{"coordinates":[ \n' ;
      for (var i = 0; i < iNumLines; i++) {
            curEOL = curPos + lineWidth ;

            rec.date = dat.slice(curPos, curPos+12);
            curPos = curPos + 12;
            rec.time = dat.slice(curPos, curPos+12);

            curPos = curPos + 12 ;

      if (curPos < totalChar) {  // Have we reached end of the file?
            rec.X = "" ;
            rec.X = extractData(dat.slice(curPos, curEOL)) ;
            curPos = curPos + rec.X.length + 1 ;

            rec.Y = "" ;
            rec.Y = extractData(dat.slice(curPos, curEOL)) ;
            curPos = curPos + rec.Y.length + 1;
```

```
            rec.Z = "" ;
            rec.Z = extractData(dat.slice(curPos, curEOL)) ;

            fileDisplayArea.innerHTML = fileDisplayArea.innerHTML +
                           "{" + '"Date":' + '"' + rec.date + '"' + ","
                               + '"Time":' + '"' + rec.time + '"' + ","
                                           + '"X":' + rec.X + ","
                                           + '"Y":' + rec.Y +  ","
                                           + '"Z":' + rec.Z  ;
            fileDisplayArea.innerHTML = fileDisplayArea.innerHTML  + "}, \n";
      } // end if

               curPos = curEOL + 1 ;
} // next
            var sTraJSON = fileDisplayArea.innerHTML.toString() ;
            var iLastCharNum = sTraJSON.length ;

            sTraJSON = sTraJSON.substring(0,iLastCharNum -3) ;
            sTraJSON = sTraJSON + "]}" ;

            fileDisplayArea.innerHTML = sTraJSON ;
}
             reader.readAsText(file) ;

} else {
            fileDisplayArea.innerText = "File not supported!"
                  }
});
}
</script>
<div id="page-wrapper">

            <h1>Trajectory Fixed Width Data File to JSON Converter</h1>
            <div>
                  Select a trajectory data file:
                  <input type="file" id="fileInput">
            </div>
            <pre id="fileDisplayArea"></pre>

      </div>
</body>
</html>
```

## Appendix B – Source code for the web-based mission visualization

```
<!DOCTYPE html>
<head>
   <meta http-equiv="X-UA-Compatible" content="chrome=1" />
   <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
<title>Trajectory from GMAT</title>
   <meta name="author" content="Daniel A. O'Neil">
   <meta name="copyright" content="? Daniel A. O'Neil" />
<style>
  p.case { clear: both; border-top: 0px; solid: #fff; }
</style>
<link rel="stylesheet" type="text/css" href="x3dom.css" />
<script src="missionfile.js" type="text/javascript"></script>
</head>
<body onload="startUpdate()">

<h1>Prototype Web Visualization of a GMAT Generated Trajectory</h1>

      <p class="case">
            <X3D xmlns="http://www.web3d.org/specifications/x3d-namespace"
showStat="false" showLog="false" x="0px" y="0px" width="800px"
height="600px">
          <Scene>
          <background DEF='bgnd' transparency='0' skyColor='0.0 0.0 0.0' >
          </background>
         <Transform id="theEarth" translation="0 0 0">
              <Shape>
                <Appearance>
                    <ImageTexture url='EarthImage.jpg'/>

                </Appearance>
                <Sphere radius='0.65'/>
             </Shape>
               </Transform>
            <Viewpoint fieldOfView="0.785398" position="6 5.5 6.5"
            orientation="1 -1 0 -0.785" description=""/>
          </Scene>
        </X3D>
    </p>

      <p class="case" align="center">

      <p id="demo"></p>

<script>

// Replace the following line with the JSON file reader.
var step = 0 ;
var JsonObj = null
var mission = null ;

function startUpdate() {
// Generate line segments from points around the trajectory.
  var segIndex = 0 ;   // segment counter
  var orbitCoords = "" ;
```

```
 function handleFileSelect(evt) {
    var files = evt.target.files; // FileList object
     f = files[0];
      var reader = new FileReader();

      // Closure to capture the file information.
      reader.onload = (function(theFile) {
        return function(e) {

      mission = JSON.parse(e.target.result);   // Parse trajectory object.

   for (var segment in mission.coordinates) {
      var s = document.createElement('Shape');            // Shape Node
      s.setAttribute("id", "segment" + segIndex);
      var app = document.createElement('Appearance');     // Appearance Node
      var mat = document.createElement('Material');       // Material Node
         mat.setAttribute("id", "Mat" + segIndex);
         mat.setAttribute("diffuseColor", 1 + " " + 0 + " " + 0);
         mat.setAttribute("emissiveColor", 1 + " " + 0 + " " + 0.3);
            app.appendChild(mat);
            s.appendChild(app);

      var segCoords = [mission.coordinates[segment].X/10000,
mission.coordinates[segment].Y/10000, mission.coordinates[segment].Z/10000] ;

orbitCoords = orbitCoords + segCoords[0]  + " " + segCoords[1]  + " " +
segCoords[2]  + " " ;

      var line = document.createElement('IndexedLineSet');
      line.setAttribute("coordIndex", segIndex);
      var coords = document.createElement('Coordinate');
      coords.setAttribute("point", orbitCoords);

      line.appendChild(coords) ;

      s.appendChild(line);
      var ot = document.getElementById('theEarth');
      ot.appendChild(s);

      segIndex = segIndex + 1 ;
       }

    /*----------------------------
     *  Create a satellite model
     *----------------------------*/
 var pos = [mission.coordinates[0].X/10000, mission.coordinates[0].Y/10000,
mission.coordinates[0].Z/10000]
  var t = document.createElement('Transform');
       t.setAttribute("translation", pos[0] + " " + pos[1] + " " + pos[2] );
       t.setAttribute("id", 'satPosition');
            t.setAttribute("scale", 0.07 + " " + 0.07 + " " + 0.07) ;

      var satModel = document.createElement('Inline') ;
      satModel.setAttribute('url', "TrianaSatellite.x3d") ;


      t.appendChild(satModel) ;
```

```
      var objsat = document.getElementById('theEarth');
        objsat.appendChild(t);

            setInterval(function () {updatePosition() }, 50);

        };
      })(f);
      reader.readAsText(f);
    }
document.getElementById('files').addEventListener('change', handleFileSelect,
false);

   };

   function updatePosition()   {
     pos = [mission.coordinates[step].X/10000,
mission.coordinates[step].Y/10000, mission.coordinates[step].Z/10000] ;
        var Xpos = pos[0] ;
        var Ypos = pos[1] ;
        var Zpos = pos[2] ;
      document.getElementById('satPosition').setAttribute('translation', Xpos
+ " " + Ypos + " " + Zpos);

document.getElementById("demo").innerHTML =
"<p>" + "<H2>" + "Date:  " + mission.coordinates[step].Date + "  Time: " +
mission.coordinates[step].Time + "</H2>" + "<p>"

 //  }
 step = step + 1 ;
 };

 function resetPosition() {
    step = 0 ;
      pos = [mission.coordinates[step].X/10000,
mission.coordinates[step].Y/10000, mission.coordinates[step].Z/10000] ;
        var Xpos = pos[0] ;
        var Ypos = pos[1] ;
        var Zpos = pos[2] ;
      document.getElementById('satPosition').setAttribute('translation', Xpos
+ " " + Ypos + " " + Zpos);

        setInterval(function () {updatePosition() }, 60);
 } ;
 </script>
<script type="text/javascript" src="x3dom.js"></script>
<input type="file" id="files" name="files[]" multiple />
<input type="button" id="reset" value="Reset" onclick="resetPosition();" />
</body>
</html>
```

# Appendix C – Trajectory data file exported from the GMAT

| Sat.UTCGregorian | Sat.EarthMJ2000Eq.X | Sat.EarthMJ2000Eq.Y | Sat.EarthMJ2000Eq.Z |
|---|---|---|---|
| 22 Jul 2014 11:29:10.811 | 137380 | -75679.6 | 21487.6 |
| 22 Jul 2014 11:30:10.811 | 137394 | -75652.8 | 21492.7 |
| 22 Jul 2014 11:33:00.381 | 137433 | -75576.9 | 21507.2 |
| 22 Jul 2014 11:41:06.513 | 137542 | -75358.2 | 21548.3 |
| 22 Jul 2014 12:07:06.014 | 137873 | -74644.6 | 21676.7 |
| 22 Jul 2014 12:52:06.014 | 138365 | -73366.1 | 21886.5 |
| 22 Jul 2014 13:37:06.014 | 138755 | -72033.8 | 22080.1 |
| 22 Jul 2014 14:22:06.014 | 139043 | -70648.3 | 22257.4 |
| 22 Jul 2014 15:07:06.014 | 139227 | -69210.5 | 22418 |
| 22 Jul 2014 15:52:06.014 | 139307 | -67720.8 | 22561.8 |
| 22 Jul 2014 16:37:06.014 | 139281 | -66179.9 | 22688.4 |
| 22 Jul 2014 17:22:06.014 | 139148 | -64588.2 | 22797.5 |
| 22 Jul 2014 18:07:06.014 | 138907 | -62946.4 | 22888.8 |
| 22 Jul 2014 18:52:06.014 | 138556 | -61254.7 | 22961.9 |
| 22 Jul 2014 19:37:06.014 | 138092 | -59513.6 | 23016.3 |
| 22 Jul 2014 20:22:06.014 | 137515 | -57723.5 | 23051.8 |
| 22 Jul 2014 21:07:06.014 | 136821 | -55884.6 | 23067.6 |
| 22 Jul 2014 21:52:06.014 | 136008 | -53997.3 | 23063.4 |
| 22 Jul 2014 22:37:06.014 | 135074 | -52061.8 | 23038.5 |
| 22 Jul 2014 23:22:06.014 | 134015 | -50078.4 | 22992.2 |
| 23 Jul 2014 00:07:06.014 | 132828 | -48047.1 | 22923.9 |
| 23 Jul 2014 00:52:06.014 | 131509 | -45968.3 | 22832.9 |
| 23 Jul 2014 01:37:06.014 | 130054 | -43842.1 | 22718.2 |
| 23 Jul 2014 02:22:06.014 | 128459 | -41668.6 | 22578.9 |
| 23 Jul 2014 03:07:06.014 | 126718 | -39448.1 | 22413.9 |
| 23 Jul 2014 03:52:06.014 | 124826 | -37180.5 | 22222.2 |
| 23 Jul 2014 04:37:06.014 | 122776 | -34866.2 | 22002.5 |
| 23 Jul 2014 05:22:06.014 | 120562 | -32505.3 | 21753.2 |
| 23 Jul 2014 06:07:06.014 | 118176 | -30098.1 | 21472.9 |
| 23 Jul 2014 06:52:06.014 | 115609 | -27644.9 | 21159.6 |
| 23 Jul 2014 07:37:06.014 | 112851 | -25146.1 | 20811.3 |
| 23 Jul 2014 08:22:06.014 | 109891 | -22602.3 | 20425.6 |
| 23 Jul 2014 09:07:06.014 | 106715 | -20014.3 | 19999.8 |
| 23 Jul 2014 09:52:06.014 | 103308 | -17383.1 | 19530.8 |
| 23 Jul 2014 10:37:06.014 | 99652.6 | -14710.3 | 19014.7 |
| 23 Jul 2014 11:22:06.014 | 95728.2 | -11997.9 | 18447.3 |
| 23 Jul 2014 12:07:06.014 | 91509.8 | -9248.86 | 17823.2 |
| 23 Jul 2014 12:52:06.014 | 86967.4 | -6467.29 | 17135.9 |
| 23 Jul 2014 13:37:06.014 | 82064.3 | -3659.21 | 16377.5 |
| 23 Jul 2014 14:22:06.014 | 76754.2 | -833.407 | 15537.9 |
| 23 Jul 2014 15:07:06.014 | 70978.2 | 1996.83 | 14603.8 |
| 23 Jul 2014 15:42:47.351 | 66015.5 | 4231.28 | 13784.5 |
| 23 Jul 2014 16:14:45.146 | 61239.5 | 6211.26 | 12981.6 |
| 23 Jul 2014 16:43:20.220 | 56663.5 | 7951.03 | 12198.9 |
| 23 Jul 2014 17:08:54.363 | 52289.8 | 9467.93 | 11438.4 |
| 23 Jul 2014 17:31:47.084 | 48117.9 | 10778.4 | 10701.4 |
| 23 Jul 2014 17:52:15.897 | 44145.3 | 11898.2 | 9988.66 |
| 23 Jul 2014 18:12:38.357 | 39935.3 | 12942.2 | 9221.22 |
| 23 Jul 2014 18:28:50.962 | 36373.4 | 13704.4 | 8561.65 |
| 23 Jul 2014 18:44:59.590 | 32609.3 | 14381.3 | 7853.71 |
| 23 Jul 2014 18:57:54.044 | 29420.7 | 14843.9 | 7244.57 |
| 23 Jul 2014 19:10:45.090 | 26063.7 | 15212 | 6593.16 |
| 23 Jul 2014 19:20:43.564 | 23314.9 | 15414.6 | 6051.37 |
| 23 Jul 2014 19:31:00.241 | 20333.1 | 15523.9 | 5454.26 |
| 23 Jul 2014 19:40:19.768 | 17479 | 15509.4 | 4872.58 |
| 23 Jul 2014 19:47:54.609 | 15041.5 | 15393.8 | 4367.03 |
| 23 Jul 2014 19:54:26.674 | 12846.8 | 15198.5 | 3904.09 |
| 23 Jul 2014 20:00:08.125 | 10858.8 | 14937.9 | 3477.63 |
| 23 Jul 2014 20:05:11.727 | 9027.54 | 14618.6 | 3078.04 |
| 23 Jul 2014 20:10:13.770 | 7143.92 | 14200.2 | 2659.39 |
| 23 Jul 2014 20:14:47.726 | 5381.99 | 13714.4 | 2259.74 |
| 23 Jul 2014 20:18:56.205 | 3742.03 | 13166.8 | 1879.65 |
| 23 Jul 2014 20:22:43.925 | 2208.59 | 12557.3 | 1515.96 |
| 23 Jul 2014 20:25:46.115 | 965.985 | 11982.2 | 1214.33 |
| 23 Jul 2014 20:28:47.087 | -275.232 | 11321.9 | 905.742 |
| 23 Jul 2014 20:31:30.966 | -1397.15 | 10637.3 | 619.35 |
| 23 Jul 2014 20:33:59.684 | -2404.93 | 9936.78 | 354.826 |
| 23 Jul 2014 20:36:17.646 | -3322.16 | 9213.97 | 106.822 |
| 23 Jul 2014 20:38:28.799 | -4169.28 | 8458.01 | -129.734 |

| | | | |
|---|---|---|---|
| 23 Jul 2014 20:40:36.645 | -4962.21 | 7654.68 | -359.289 |
| 23 Jul 2014 20:42:44.536 | -5712.84 | 6785.65 | -585.814 |
| 23 Jul 2014 20:44:55.140 | -6424.59 | 5833.41 | -811.489 |
| 23 Jul 2014 20:47:05.204 | -7067.92 | 4825.77 | -1027.94 |
| 23 Jul 2014 20:49:10.008 | -7615.52 | 3811.11 | -1225.49 |
| 23 Jul 2014 20:51:12.320 | -8080.33 | 2780.67 | -1407.52 |
| 23 Jul 2014 20:53:12.395 | -8464.65 | 1744.18 | -1573.67 |
| 23 Jul 2014 20:55:11.875 | -8775.96 | 698.077 | -1725.76 |
| 23 Jul 2014 20:57:16.831 | -9027.72 | -401.824 | -1870.28 |
| 23 Jul 2014 20:59:36.631 | -9224.96 | -1628.4 | -2014.43 |
| 23 Jul 2014 21:01:54.390 | -9339.41 | -2823.22 | -2139.03 |
| 23 Jul 2014 21:04:06.968 | -9382.49 | -3953.1 | -2243.58 |
| 23 Jul 2014 21:06:20.802 | -9367.1 | -5069.01 | -2335.03 |
| 23 Jul 2014 21:08:39.894 | -9296.47 | -6199.26 | -2416.45 |
| 23 Jul 2014 21:11:07.144 | -9169.45 | -7361.01 | -2489.06 |
| 23 Jul 2014 21:13:45.296 | -8982.27 | -8568.04 | -2553.25 |
| 23 Jul 2014 21:16:37.225 | -8728.98 | -9832.71 | -2608.87 |
| 23 Jul 2014 21:19:45.879 | -8402.04 | -11165.2 | -2655.23 |
| 23 Jul 2014 21:23:13.639 | -7994.19 | -12569 | -2691.17 |
| 23 Jul 2014 21:27:01.532 | -7501.3 | -14037.4 | -2715.27 |
| 23 Jul 2014 21:31:09.956 | -6922.04 | -15559.6 | -2726.34 |
| 23 Jul 2014 21:35:40.529 | -6253.4 | -17132.7 | -2723.53 |
| 23 Jul 2014 21:40:36.622 | -5488.39 | -18763 | -2705.96 |
| 23 Jul 2014 21:46:03.170 | -4615.81 | -20462.4 | -2672.34 |
| 23 Jul 2014 21:52:37.252 | -3535.53 | -22391.4 | -2615.91 |
| 23 Jul 2014 21:59:31.570 | -2379.75 | -24293.7 | -2541.85 |
| 23 Jul 2014 22:07:21.455 | -1055.71 | -26315.5 | -2443.66 |
| 23 Jul 2014 22:15:56.177 | 400.475 | -28386.8 | -2322.78 |
| 23 Jul 2014 22:25:19.625 | 1992.47 | -30505.7 | -2178.29 |
| 23 Jul 2014 22:35:37.509 | 3728.27 | -32675.4 | -2008.84 |
| 23 Jul 2014 22:46:56.246 | 5616.8 | -34898.8 | -1812.87 |
| 23 Jul 2014 22:59:22.911 | 7667.53 | -37178.1 | -1588.62 |
| 23 Jul 2014 23:13:05.305 | 9890.4 | -39514.5 | -1334.18 |
| 23 Jul 2014 23:28:12.026 | 12295.6 | -41908.4 | -1047.5 |
| 23 Jul 2014 23:44:52.571 | 14893.8 | -44359.2 | -726.394 |
| 24 Jul 2014 00:03:17.418 | 17695.6 | -46865.3 | -368.534 |
| 24 Jul 2014 00:23:38.154 | 20711.7 | -49423.6 | 28.5123 |
| 24 Jul 2014 00:46:07.611 | 23953 | -52030.2 | 467.29 |
| 24 Jul 2014 01:10:59.980 | 27430.1 | -54679.2 | 950.426 |
| 24 Jul 2014 01:38:30.971 | 31153.1 | -57363.4 | 1480.61 |
| 24 Jul 2014 02:08:57.952 | 35131.5 | -60073.1 | 2060.57 |
| 24 Jul 2014 02:42:40.146 | 39373.7 | -62796.9 | 2693.02 |
| 24 Jul 2014 03:19:58.987 | 43887.2 | -65520.4 | 3380.66 |
| 24 Jul 2014 04:01:18.042 | 48677.4 | -68226.2 | 4126.03 |
| 24 Jul 2014 04:46:18.042 | 53665.1 | -70851.7 | 4918.35 |
| 24 Jul 2014 05:31:18.042 | 58431.5 | -73184.7 | 5690.38 |
| 24 Jul 2014 06:16:18.042 | 62992.1 | -75259.7 | 6442.33 |
| 24 Jul 2014 07:01:18.042 | 67360.2 | -77104.6 | 7174.56 |
| 24 Jul 2014 07:46:18.042 | 71547.2 | -78742 | 7887.44 |
| 24 Jul 2014 08:31:18.042 | 75562.8 | -80190.8 | 8581.38 |
| 24 Jul 2014 09:16:18.042 | 79415.6 | -81466.9 | 9256.79 |
| 24 Jul 2014 10:01:18.042 | 83113.3 | -82583.8 | 9914.04 |
| 24 Jul 2014 10:46:18.042 | 86662.3 | -83553.1 | 10553.5 |
| 24 Jul 2014 11:31:18.042 | 90068.5 | -84384.8 | 11175.5 |
| 24 Jul 2014 12:16:18.042 | 93337.2 | -85087.7 | 11780.4 |
| 24 Jul 2014 13:01:18.042 | 96473 | -85669.5 | 12368.4 |
| 24 Jul 2014 13:46:18.042 | 99480 | -86137 | 12939.9 |
| 24 Jul 2014 14:31:18.042 | 102362 | -86496.3 | 13495 |
| 24 Jul 2014 15:16:18.042 | 105122 | -86752.9 | 14034 |
| 24 Jul 2014 16:01:18.042 | 107763 | -86911.5 | 14557.1 |
| 24 Jul 2014 16:46:18.042 | 110289 | -86976.6 | 15064.4 |
| 24 Jul 2014 17:31:18.042 | 112700 | -86952.2 | 15556.1 |
| 24 Jul 2014 18:16:18.042 | 115000 | -86841.8 | 16032.3 |
| 24 Jul 2014 19:01:18.042 | 117191 | -86648.8 | 16493.2 |
| 24 Jul 2014 19:46:18.042 | 119273 | -86376 | 16938.7 |
| 24 Jul 2014 20:31:18.042 | 121249 | -86026.3 | 17369.2 |
| 24 Jul 2014 21:16:18.042 | 123120 | -85602.2 | 17784.4 |
| 24 Jul 2014 22:01:18.042 | 124887 | -85106 | 18184.6 |
| 24 Jul 2014 22:46:18.042 | 126552 | -84539.8 | 18569.7 |
| 24 Jul 2014 23:31:18.042 | 128114 | -83905.6 | 18939.8 |
| 25 Jul 2014 00:16:18.042 | 129575 | -83205.1 | 19294.8 |
| 25 Jul 2014 01:01:18.042 | 130935 | -82440.1 | 19634.7 |

| | | | |
|---|---|---|---|
| 25 Jul 2014 01:46:18.042 | 132194 | -81612.1 | 19959.5 |
| 25 Jul 2014 02:31:18.042 | 133354 | -80722.5 | 20269.1 |
| 25 Jul 2014 03:16:18.042 | 134414 | -79772.7 | 20563.5 |
| 25 Jul 2014 04:01:18.042 | 135375 | -78763.8 | 20842.6 |
| 25 Jul 2014 04:46:18.042 | 136235 | -77697.1 | 21106.2 |
| 25 Jul 2014 05:31:18.042 | 136996 | -76573.5 | 21354.2 |
| 25 Jul 2014 06:16:18.042 | 137657 | -75394 | 21586.6 |
| 25 Jul 2014 07:01:18.042 | 138217 | -74159.6 | 21803.1 |
| 25 Jul 2014 07:46:18.042 | 138676 | -72871.1 | 22003.5 |
| 25 Jul 2014 08:31:18.042 | 139033 | -71529.2 | 22187.7 |
| 25 Jul 2014 09:16:18.042 | 139287 | -70134.7 | 22355.5 |
| 25 Jul 2014 10:01:18.042 | 139438 | -68688.2 | 22506.6 |
| 25 Jul 2014 10:46:18.042 | 139484 | -67190.3 | 22640.7 |
| 25 Jul 2014 11:29:10.811 | 139430 | -65715.8 | 22752.4 |