Developing Automated Geospatial Procedures in Civil Engineering

Joseph Kaelin - Katharina Kaelin - Philippe Nater - Alan Hodgkinson
Civil Engineering
Motivation

**low productivity** in Civil Engineering

dominance of ad hoc procedures and lack of standard procedures

**automation and reuse**

coding is a good way to automate and push standard procedures
Automated Procedures for Civil Engineering

**aim?**
- automate routine work
- not aimed at replacing engineer
- most tasks are routine
- use standard design criteria & methodology
- free engineers to add project-specific value

**need?**
- feasibility investigations
- alternative studies
- design & documentation
- optimization
Data Organizer - rock lab data (GeoPython 2017)
## Decision Support System

<table>
<thead>
<tr>
<th>integrated data</th>
<th>domain-specific procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>select, document, code engineering procedures</td>
</tr>
<tr>
<td></td>
<td>- provides substantial time-savings</td>
</tr>
<tr>
<td></td>
<td>- improves quality control</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>selection criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>- ‘fitness for purpose’</td>
</tr>
<tr>
<td>- building codes</td>
</tr>
<tr>
<td>- industry standards</td>
</tr>
<tr>
<td>- cost</td>
</tr>
</tbody>
</table>
Initial Focus is on Hydropower Waterways

‘what-if’ analysis & optimization
alternative alignments, sizing, construction methods
variations in geological forecast

selection criteria
hydraulics, geology, leakage, constructibility, cost, construction time
Input Data

Terrain DTM
- hillslopes, watersheds, stream courses

Groundwater surface

Rock surface
- geological parameters

Alignments
- layout of project features

better organisation and visualization of input data
<table>
<thead>
<tr>
<th>tunnel layout</th>
<th>plan views (map)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>longitudinal section (profile)</td>
</tr>
<tr>
<td>tunnel cross-sections</td>
<td>excavation</td>
</tr>
<tr>
<td></td>
<td>rock support</td>
</tr>
<tr>
<td></td>
<td>quantities</td>
</tr>
<tr>
<td></td>
<td>python CAD?</td>
</tr>
<tr>
<td>result plots &amp; tables</td>
<td></td>
</tr>
<tr>
<td>work schedules</td>
<td>time-distance diagrams</td>
</tr>
</tbody>
</table>
Minimum Viable Product

proof of concept

just enough to see if it works
Focus on Hydropower Waterways

unlined pressures tunnels  - widespread use due to economics
                             - requires rock mass to be tight against leakage
                             - concrete lining, steel lining in short sections

design criteria for hydraulic confinement
                             - prevent opening of joints
Civil Engineering
Design Criteria Hydraulic Confinement

Norwegian criteria selected as best practice
- confinement assumed related to overburden stress
- design validation requires hydraulic jacking tests

\[ C_{RM} = \frac{h_s \gamma_w F}{\gamma_R \cos \beta} \]

- \( C_{RM} \) = MINIMUM ROCK COVER
- \( h_s \) = STATIC HEAD
- \( \gamma_w \) = UNIT WEIGHT OF WATER
- \( \gamma_R \) = UNIT WEIGHT OF ROCK
- \( \beta \) = SLOPE ANGLE (VARIES ALONG SLOPE)
- \( F \) = SAFETY FACTOR
Design Procedure Hydraulic Confinement

Current practice:
- Draw cross-sections, longitudinal sections at representative tunnel locations.
- Determine $C_{RM}$, $\beta$ at point on surface closest to tunnel.
- Calculate safety factor F in spreadsheet.
Implementation

Python

at buffer points along stationed alignment
- select buffer point with least rock cover $C_R$
- calculate safety factor $F$
Python Ecosystem for Implementation

- pyqgis processing.runalg with grass
  - using runalg vs grass alone?
- pandas, geopandas, numpy
- plotly
- (couchbase)
- rock density
Results for Hydraulic Confinement - plan view
Results for Hydraulic Confinement - profile
Improving User Experience

browser dashboard

practical way to make project spatial data available
- for use at construction sites
- for engineers who won’t use Jupyter, python
- visualization for discussions, client presentations
Browser Dashboard - Implementation

progressive web app (PWA) using Polymer web components
- plan view (map): Google Maps API
- profiles: plotly
- results plots: plotly

build API’s from procedures in Jupyter Notebook
- moving procedures to API requires robust error checking
- using API for common procedures bolsters using Notebook as design report

python procedures accessed by Polymer from Flask Server
Profile - plotly

Overburden

hydraulic confinement

What-if?

update

alignment

rockcover
Polymor and Flask Server Implementation

Coordinate Converter

epsg
2056

EPSG  epsg.io

easting
2614950

northing
1265000

CONVERT

WGS 84 Longitude/Latitude

lat: 47.535560305636466
long: 7.637168802083274
Polymer Web Components

```html
<iron-ajax
  url="http://gis.jkaelin.com:nnnn/coordinates"
  id="postExample"
  handle-as="json"
  content-type="application/json"
  method="POST"
  params='{{postBody}}'
  on-response="serverSuccess"
  last-response='{{serverSuccessObj}}'
  on-error="serverError"
  last-error='{{serverErrorObj}}'>
</iron-ajax>
</template>

<script>
class MyView1 extends Polymer.Element {
  static get is() { return 'my-view1'; }

  static get properties() {
    return {
      epsg: {
        type: String,
        notify: true,
        value:'2683418',
      },
    },

```
@app.route('/coordinates', methods=['POST', 'GET'])
def pyproj_test():
    # .... variable initialization deleted
    easting = request.values['easting']
    northing = request.values['northing']
    epsg = request.values['epsg']
    easting = request.values['easting'].split(',')
    northing = request.values['northing'].split(',')

    inProj = pp.Proj(init='epsg:'+str(epsg))
    outProj = pp.Proj(init='epsg:4326')  # WGS84

    try:
        for i in range(len(easting)):
            x1.append(float(easting[i]))
            y1.append(float(northing[i]))
            lnglat.append(pp.transform(inProj, outProj, float(easting[i]), float(northing[i])))

        return jsonify({'lng': [item[0] for item in lnglat], 'lat': [item[1] for item in lnglat]})
    except:
        pass

    return jsonify({'entry': 'invalid'})
Thank you for listening!

Thanks to Python and Polymer open source communities
Thanks also to Pöyry colleagues for support and feedback