Mathematical research data

Lars Kastner

Technische Universität Berlin

2024-07-03 OSCAR Boot Camp

Reproducibility in Mathematics

A simple experiment

Take an article from $1, 5, 10$ years ago that uses software and:

- Find the code that the authors used.
- Find the input and output data that the authors used/produced.
- Install the software that the authors used.
- Rerun the experiment.
- Compare the results to the results in the article.
- `https://polymake.org/doku.php/workshops/workshop1122/reproducibility`

We will today mostly focus on the data and software parts.

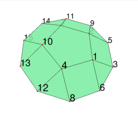
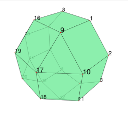
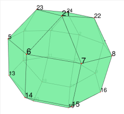
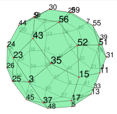
The Johnson solids: Definition

Definition

A *Johnson solid* is a

- (convex) polytope,
- 3-dimensional,
- all facets are regular polygons,
- not uniform (e.g. not Platonic, Archimedian...)

Examples

The Johnson solids: Definition

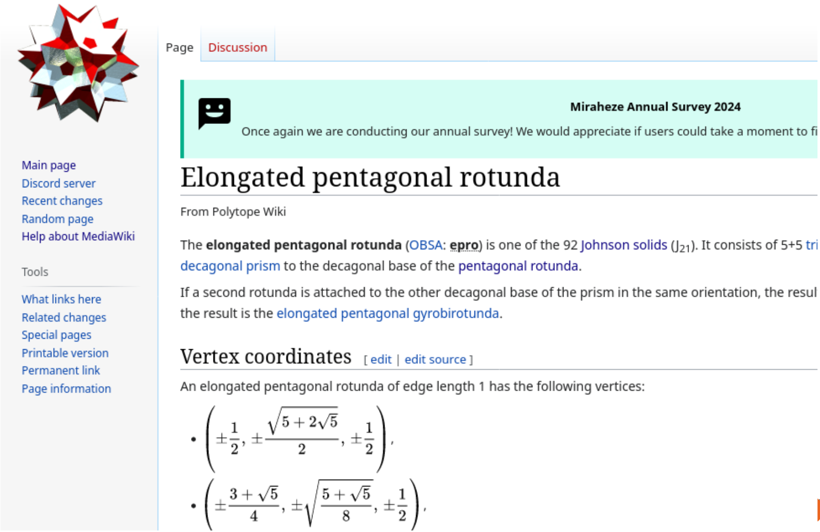
Definition

A *Johnson solid* is a

- (convex) polytope,
- 3-dimensional,
- all facets are regular polygons,
- not uniform (e.g. not Platonic, Archimedian...)

There are only finitely many Johnson solids. How can we get this data collection?

The Johnson solids: Miraheze



Main page
Discord server
Recent changes
Random page
Help about MediaWiki

Tools

What links here
Related changes
Special pages
Printable version
Permanent link
Page information

Page | Discussion

Miraheze Annual Survey 2024

Once again we are conducting our annual survey! We would appreciate if users could take a moment to fi

Elongated pentagonal rotunda

From Polytope Wiki

The **elongated pentagonal rotunda** (OBSA: epro) is one of the 92 Johnson solids (J_{21}). It consists of 5+5 tri
decagonal prism to the decagonal base of the pentagonal rotunda.

If a second rotunda is attached to the other decagonal base of the prism in the same orientation, the resul
the result is the elongated pentagonal gyrobirotunda.

Vertex coordinates [edit | edit source]

An elongated pentagonal rotunda of edge length 1 has the following vertices:

- $\left(\pm\frac{1}{2}, \pm\frac{\sqrt{5+2\sqrt{5}}}{2}, \pm\frac{1}{2}\right)$,
- $\left(\pm\frac{3+\sqrt{5}}{4}, \pm\sqrt{\frac{5+\sqrt{5}}{8}}, \pm\frac{1}{2}\right)$,

The Johnson solids: Wikipedia

WIKIPEDIA The Free Encyclopedia

Search Wikipedia | Search

Elongated pentagonal rotunda

14 languages

Article Talk | Read Edit View history Tools

Contents hide

From Wikipedia, the free encyclopedia

In geometry, the **elongated pentagonal rotunda** is one of the Johnson solids (J_{21}). As the name suggests, it can be constructed by elongating a pentagonal rotunda (J_6) by attaching a decagonal prism to its base. It can also be seen as an elongated pentagonal orthobirotunda (J_{42}) with one pentagonal rotunda removed.

A Johnson solid is one of 92 strictly convex polyhedra that is composed of regular polygon faces but are not uniform polyhedra (that is, they are not Platonic solids, Archimedean solids, prisms, or antiprisms). They were named by Norman Johnson, who first listed these polyhedra in 1966.[1]

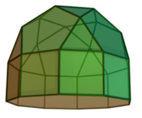
Formulae [edit]

The following formulae for volume and surface area can be used if all faces are regular, with edge length a:[2]

$$V = \frac{1}{12}\left(45 + 17\sqrt{5} + 30\sqrt{5+2\sqrt{5}}\right) a^3 \approx 14.612...a^3$$

$$A = \frac{1}{2}\left(20 + \sqrt{5\left(145 + 58\sqrt{5} + 2\sqrt{30\left(65 + 29\sqrt{5}\right)}\right)}\right) a^2 \approx 32.3472...a^2$$

Elongated pentagonal rotunda



Type	Johnson J_{20} - $\mathbf{J_{21}}$ - J_{22}
Faces	2x5 triangles 2x5 squares 1+5 pentagons 1 decagon
Edges	55
Vertices	30
Vertex configuration	$10(4^2.10)$ $10(3.4^2.5)$ $2.5(3.5.3.5)$

The Johnson solids: polymake

polymake implementation

```
dispatcher_t dispatcher[]={
   { "square_pyramid",                      &square_pyramid },
   { "pentagonal_pyramid",                  &pentagonal_pyramid },
   { "triangular_cupola",                   &triangular_cupola },
   { "square_cupola",                       &square_cupola },
   { "pentagonal_cupola",                   &pentagonal_cupola },
   { "pentagonal_rotunda",                  &pentagonal_rotunda },
   { "elongated_triangular_pyramid",        &elongated_triangular_pyramid },
   { "elongated_square_pyramid",            &elongated_square_pyramid },
   { "elongated_pentagonal_pyramid",        &elongated_pentagonal_pyramid },    //inexact
   { "gyroelongated_square_pyramid",        &gyroelongated_square_pyramid },    //inexact
   { "gyroelongated_pentagonal_pyramid",    &gyroelongated_pentagonal_pyramid },
   { "triangular_bipyramid",                &triangular_bipyramid },
   { "pentagonal_bipyramid",                &pentagonal_bipyramid },            //inexact
   { "elongated_triangular_bipyramid",      &elongated_triangular_bipyramid },
   { "elongated_square_bipyramid",          &elongated_square_bipyramid },
   { "elongated_pentagonal_bipyramid",      &elongated_pentagonal_bipyramid },  //inexact
   { "gyroelongated_square_bipyramid",      &gyroelongated_square_bipyramid },  //inexact
   { "elongated_triangular_cupola",         &elongated_triangular_cupola },     //inexact
```

The Johnson solids as data

History

- There are 92 Johnson solids (conjectured by Norman Johnson 1966).
- Proof by Victor Zalgaller.
- No coordinates.
- Errors in coordinates on `Wikipedia`.
- Errors in coordinates on `https://polytope.miraheze.org/wiki/Johnson_solid`

What is the problem?

The Johnson solids as data

History

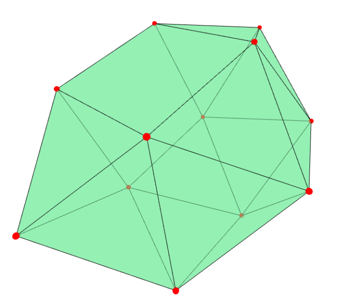
- There are 92 Johnson solids .
- Errors in coordinates on various wikis.

What is the problem?

- Most discrete mathematics software uses at most GMP rationals (i.e. $\mathbb{Q}$).
- Even quadratic field extensions are not enough.
- The Johnson solids need complicated fields to be realized properly.
- Thus, most calculations have to be done by hand or by moving between systems, which is error-prone.

The Johnson solids: Sphenomegacorona

Sphenomegacorona



The Johnson solids: Sphenomegacorona

Sphenomegacorona

```
julia> js = johnson_solid(88);

julia> k = coefficient_field(js);

julia> embedding(k)
Complex embedding corresponding to root 0.80
  of relative number field with defining polynomial y^2 + a^2 - 1
    over number field with defining polynomial 1680*x^16 - 4800*x^15 - 3712*x^14 +
           17216*x^13
    + 1568*x^12 - 24576*x^11 + 2464*x^10 + 17248*x^9 - 3384*x^8 - 5584*x^7 + 2000*x^6
           + 240*x^
    5 - 776*x^4 + 304*x^3 + 200*x^2 - 56*x - 23
      over rational field
  extending complex embedding corresponding to 0.59 of number field
```

The Johnson solids in OSCAR

How does this work?

The Johnson solids in OSCAR

How does this work? Data should be FAIR!

FAIR

The FAIR principles

- **Findable** A researcher who does not know about your data, but is working on something related to your data, can **F**ind it.
- **Accessible** A researcher who knows about your data can **A**ccess it, ideally in some standardized way, say web portal.
- **Interoperable** Different systems and applications can **I**nteract with your data, integrating it with other datasets and tools in a meaningful way.
- **Reusable** A researcher can **R**epurpose your data for new analyses or projects, ensuring that your data remains valuable over time.

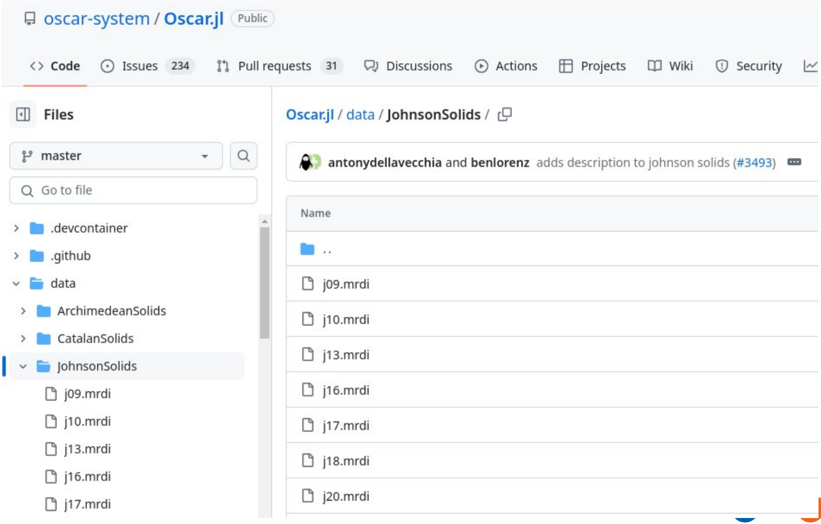
These principles also affect metadata.

OSCAR's and Julia's features for FAIRness

What features from Julia and OSCAR can help?

- OSCAR's mrdi file format
- Julia's package manager
- Julia's Project.toml and Manifest.toml files

The Johnson solids in OSCAR



The `mrdi` file format

A FAIR file format

- File format developed in MaRDI for mathematical data. [VJL23]
- Based on JSON.
- Uses `OSCAR` as a testbed.
- Many data types of `OSCAR` (and `Julia`) can already be serialized and deserialized in this format.
- Work in progress for reading this file format in CoCoA, Magma, and Sage.

The Johnson solids: Sphenomegacorona


```
{
  "_ns": {
    "Oscar": [
      "https://github.com/oscar-system/Oscar.jl"
            ,
      "1.1.0-DEV-393962
            ae3172ace5e2d4c50ef17cb8ee89828d1e"
    ]
  },
  "_type": {
    "name": "Polyhedron",
    "params": "a65b5810-9f39-4bfe-8e4d-9
          f5bf89e63b3"
  },
  "data": { [...]
        "VERTICES": {
          "name": "MatElem",
          "params": "73744d5a-9d44-49f6-a4d5-
                d9e74803139e"
        }, [...]
    "data": "VERTICES" : [[ ... some coordinates
             ... ]]
  },
```

```
{
  "_refs": { [...]
    "a65b5810-9f39-4bfe-8e4d-9f5bf89e63b3": {
      "_type": "EmbeddedNumField",
      "data": {
        "num_field": "eea1dea1-bd4c-4869-bece-
              bbf412869a55",
        "embedding": "1d972617-9785-4f3d-8769-02
              ecd7fedc25"
      }
    },
    "73744d5a-9d44-49f6-a4d5-d9e74803139e": {
      "_type": "MatSpace",
      "data": {
        "base_ring": "a65b5810-9f39-4bfe-8e4d-9
              f5bf89e63b3",
        "ncols": "4",
        "nrows": "12"
      }
    },
    [...]
```

MaRDI

- **Ma**thematical **R**esearch **D**ata **I**nitiative
- Funded by DFG Project number 460135501
- `https://www.mardi4nfdi.de`
- 7 Task Areas, Task Area 1: Computer Algebra
- Strong focus on reproducibility of computer experiments.
- Whitepaper on data management [Con23].

Reproducibility in Mathematics

Study from computational physics

- Victoria Stodden, Matthew S. Krafczyk, and Adhithya Bhaskar: **Enabling the Verification of Computational Results: An Empirical Evaluation of Computational Reproducibility** (2018) [SKB18]
- 306 papers from the Journal of Computational Physics were surveyed.
- More than half of the results were *impossible* to reproduce.
- None of the results could be reproduced with minimal effort.

Reproducibility in Mathematics

Study from a SFB of applied mathematics

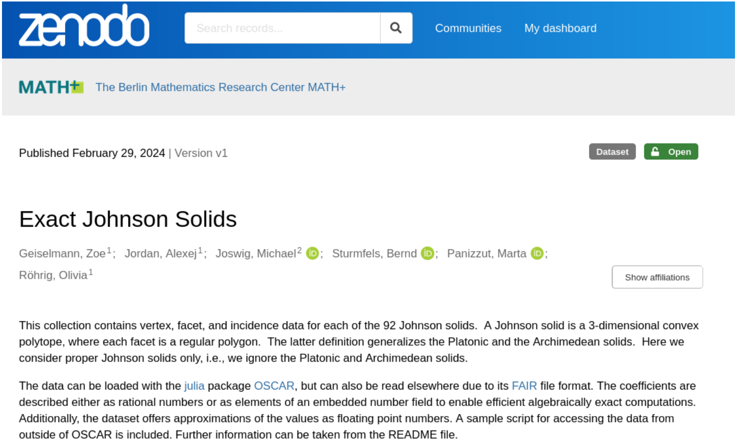
- Christian Riedel et al.: **Including Data Management in Research Culture Increases the Reproducibility of Scientific Results** (2022) [Rie+22]
- This study "analyzes the reproducibility of 108 publications from an interdisciplinary Collaborative Research Center on applied mathematics in various scientific fields."
- Roughly 40% of the publications had enough data for attempting replication at all
- Only 4 were considered "fully reproducible".

Reproducibility in Mathematics

Study of reproducibility of Jupyter notebooks

- Daniel Mietchen; Sheeba Samuel: **Computational reproducibility of Jupyter notebooks from biomedical publications** (2022) [SM22]
- Jupyter notebooks allow for highly automated testing.
- More than 10 000 Jupyter notebooks from biomedical publications were analyzed.
- Roughly 4 000 notebooks were selected for a reproducibility check.
- Of these only 10% could be rerun.
- Only 6% actually resulted in the recorded results.

https://zenodo.org/records/10729583



zenodo Search records... Communities My dashboard

MATH+ The Berlin Mathematics Research Center MATH+

Published February 29, 2024 | Version v1

Dataset Open

Exact Johnson Solids

Geiselmann, Zoe[1]; Jordan, Alexej[1]; Joswig, Michael[2]; Sturmfels, Bernd; Panizzut, Marta; Röhrig, Olivia[1]

Show affiliations

This collection contains vertex, facet, and incidence data for each of the 92 Johnson solids. A Johnson solid is a 3-dimensional convex polytope, where each facet is a regular polygon. The latter definition generalizes the Platonic and the Archimedean solids. Here we consider proper Johnson solids only, i.e., we ignore the Platonic and Archimedean solids.

The data can be loaded with the julia package OSCAR, but can also be read elsewhere due to its FAIR file format. The coefficients are described either as rational numbers or as elements of an embedded number field to enable efficient algebraically exact computations. Additionally, the dataset offers approximations of the values as floating point numbers. A sample script for accessing the data from outside of OSCAR is included. Further information can be taken from the README file.

Zenodo

- Launched in 2013, relaunched in 2015.
- Operated by CERN.
- Up to 50GB per dataset.
- Provides DOI to dataset and BibTeX export for citation.
- Can integrate github repositories.

The OSCAR book

- Consists of 19 chapters.
- Every chapter contains code.
- How can we make sure that the code keeps working?
- How can we guarantee that the output in the book is the same as in OSCAR 1.0?

OscarBookExamples

Build Status

To test some parts of the book please try the following:

Clone this repository, the oscar-book repository and a create a checkout of the Oscar.jl repo on the `backports-release-1.0` branch.

```
$ julia-1.10
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.10.1 (2024-02-13)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |
```

OscarBookExamples.jl

- Collects the code from the `OSCAR` book.
- Runs it with the current master of `OSCAR` (other branches are possible).
- Compares output, detects errors.
- Can automatically fix the code in the book.

OscarBookExamples.jl

Oscar.jl / test / book / cornerstones / **polyhedral-geometry** /

lkastner and **benlorenz** Add tests of book chapter to CI. (#3588)

Name	Last commit message
..	
auxiliary_code	Add tests of book chapter to CI. (#3588)
D222Computation.jlcon	Add tests of book chapter to CI. (#3588)
Explosion.jlcon	Add tests of book chapter to CI. (#3588)
GKZ_orbits.jlcon	Add tests of book chapter to CI. (#3588)
GT_character.jlcon	Add tests of book chapter to CI. (#3588)
GelfandTsetlinEx.jlcon	Add tests of book chapter to CI. (#3588)
SecondaryPolytope.jlcon	Add tests of book chapter to CI. (#3588)
ch-benchmark.jlcon	Add tests of book chapter to CI. (#3588)
dodecahedron.jlcon	Add tests of book chapter to CI. (#3588)

OscarBookExamples.jl

- 90% of examples give different output than recorded in the book.
- 70% resulted in errors.
- Managed to bring these numbers down to 0 with `OSCAR` 1.0.
- Input is intended to keep working with `OSCAR` 1.x, output allowed to change for ¿1.0.

GitHub

- Launched in 2008.
- Bought by Microsoft in 2018.
- For code, provides easy CI integration.
- Every repository is a git.
- Not suitable for all kinds of data, data should be "git compatible", i.e. small changes of the data should be reflected as "small" changes of the files.

Project.toml

- Records all (direct) dependencies of a project.
- Contained in any Julia project.
- Also lists version requirements.
- Has metadata of project.

```
name = "Oscar"
uuid = "f1435218-dba5-11e9-1e4d-f1a5fab5fc13"
authors = ["The OSCAR Team <oscar@oscar-system.
      org>"]
version = "1.2.0-DEV"

[deps]
AbstractAlgebra = "c3fe647b-3220-5bb0-a1ea-
      a7954cac585d"
AlgebraicSolving = "66b61cbe-0446-4d5d-9090-1
      ff510639f9d"
Distributed = "8ba89e20-285c-5b6f-9357-94700520
      ee1b"
GAP = "c863536a-3901-11e9-33e7-d5cd0df7b904"
   [...]

[compat]
AbstractAlgebra = "0.41.3"
AlgebraicSolving = "0.4.15"
Distributed = "1.6"
GAP = "0.10.2"
   [...]
```

Manifest.toml

- Recursively records all dependencies.
- Concrete versions used.
- `status --manifest`
- `]activate`
- `]instantiate`

```
# This file is machine-generated - editing it
      directly is not advised

julia_version = "1.10.4"
manifest_format = "2.0"
project_hash = "071
      e3917b23a0c93525d24c6bdd638fec791b78f"

[[deps.ASL_jll]]
deps = ["Artifacts", "JLLWrappers", "Libdl", "
      Pkg"]
git-tree-sha1 = "6252039
      f98492252f9e47c312c8ffda0e3b9e78d"
uuid = "ae81ac8f-d209-56e5-92de-9978fef736f9"
version = "0.1.3+0"

[[deps.AbstractAlgebra]]
deps = ["InteractiveUtils", "LinearAlgebra", "
      MacroTools", "Preferences", "Random", "
      RandomExtensions", "SparseArrays", "Test"
      ]
git-tree-sha1 = "6338
      a830da4d86d107c906971e44f70e3148b9cb"
uuid = "c3fe647b-3220-5bb0-a1ea-a7954cac585d"
version = "0.41.9"

[[deps.Adapt]]
deps = ["LinearAlgebra", "Requires"]
```

The end

Thank you!

[Con23] The MaRDI Consortium. *Research Data Management Planning in Mathematics*. Oct. 2023. DOI: 10.5281/zenodo.10018246.

[Gei+24] Zoe Geiselmann et al. *Exact Johnson Solids*. Zenodo, Mar. 2024. DOI: 10.5281/zenodo.10729583.

[Rie+22] Christian Riedel et al. "Including Data Management in Research Culture Increases the Reproducibility of Scientific Results". In: *GI-Jahrestagung*. 2022.

[SKB18] Victoria Stodden, Matthew S. Krafczyk, and Adhithya Bhaskar. "Enabling the Verification of Computational Results: An Empirical Evaluation of Computational Reproducibility". In: *Proceedings of the First International Workshop on Practical Reproducible Evaluation of Computer Systems*. P-RECS'18. Tempe, AZ, USA: Association for Computing Machinery, 2018. DOI: 10.1145/3214239.3214242.

[SM22] Sheeba Samuel and Daniel Mietchen. *Computational reproducibility of Jupyter notebooks from biomedical publications*. 2022. arXiv: 2209.04308 [cs.CE].

[VJL23] Antony Della Vecchia, Michael Joswig, and Benjamin Lorenz. *A FAIR File Format for Mathematical Software*. 2023. arXiv: 2309.00465 [cs.MS].