INFORMAČNÍ BULLETIN

České statistické společnosti

Ročník 31, číslo 3, září 2020

Obsah

Zprávy a informace

Informační bulletin České statistické společnosti vychází čtyřikrát do roka v českém vydání. Příležitostně i mimořádné české a anglické číslo. Vydavatelem je Česká statistická společnost, IČ 00550795, adresa společnosti je Na padesátém 81, 100 82 Praha 10. Evidenční číslo registrace vedené Ministerstvem kultury ČR dle zákona č. 46/2000 Sb. je E 21214. Časopis je sázen v programu TEX, ve formátu LuaHBTEX s písmy balíku CSfonts.

The Information Bulletin of the Czech Statistical Society is published quarterly.
The contributions in the journal are published in English, Czech and Slovak languages.

DOI: 10.5300/IB, http://dx.doi.org/10.5300/IB
ISSN 1210–8022 (Print), ISSN 1804–8617 (Online)

Toto číslo bylo vytištěno s laskavou podporou Českého statistického úřadu.

NOVINKY ZE SVĚTA R+KORONAVIRU
NEWS FROM THE WORLD OF R+CORONAVIRUS

Pavel Stříž

E-mail: pavel@striz.cz

The R Foundation Retweeted: Peter Dalgaard. R 4.0.0 "Arbor Day" (source version) has been released.

24. 4. 2020 mi přistála na stole tato zpráva a za pár dní na to, 28. 4. 2020, došlo k aktualizaci Bioconductoru na verzi 3.11. Je Svátek práce, jdu ty nové `lazyLoad`, `lazyEval` a `lazyData` v R vyzkoušet a sdělit svůj nezávislý pohled.

1. R v4.0.0

Bez otálení jsem na svém Xubuntu 18.04 do `/etc/apt/sources.list` přidal:

```
deb https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/
```

Zakomentoval jsem starší pokusy a provedl preventivní kroky:

```
$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
    E298A3A825C0D65DFD57CBB651716619E084DAB9
$ sudo apt update
$ sudo apt upgrade
```

Jádro aktualizace pak tvořil příkaz:

```
$ sudo apt install r-base r-base-dev
```

Rychlý test prokázal, že se podařilo a provedl jsem aktualizaci knihoven:

```
$ R --version
$ R
> update.packages(.libPaths())
```

2. COVID v19

The R Foundation Retweeted: R Consortium has started new GitHub repository to centralize collaboration and data sources – looking to develop COVID-19 tools and code – Come add your information and contribute to the community! https://bddy.me/3aAXOmb

Z toho samého dne zaujala mou pozornost ještě tato zpráva. Řekl jsem si, že bych měl nově nainstalované R hlouběji vyzkoušet.

Na úvodní stránce na mne vyskočily 4 projekty: Coronavirus Tracker, COVID-19 Propagation, COVID-19 Tracker Map a COVID-19 Projections.

Vezmu-li to od konce. U Projections na mne vyskočil GitHub. Po určitém bádání se mi podařilo otevřít rozcestník a webovou stránku pro Českou republiku, volzinnovation.com/covid-19_SARS-CoV-2_corona/reports/latest/Czechia.html. Autorem je Raphael Volz.

Autorem Tracker Map je Jay Ulfelder. V RStudiu Cloud mne hned upozornili, že se jedná o dočasný projekt. Pod Shiny app na mne vedle analýz vyškočily pdf v záložce WHO Situation Reports. Zajímavý nápad.

Druhý projekt v pořadí je Propagation. Autorem je Juan Francisco Venegas Gutiérrez. Bohužel repozitář na GitHubu mi nešel otevřít, tak jsem to zahlásil (Issues). Mezi modely jsem Českou republiku nenašel.

3. Coronavirus Tracker v0.1.4

První projekt v pořadí od Johna Coeneho Coronavirus Tracker vyzývá ke spuštění R. Pustil jsem se do toho. RStudio cloud občas jel bez přístupových práv, požadavek na knihovny `shinyMobile` a `echarts4r` zněl zajímavě.

```
$ R
> install.packages("coronavirus")
```

Zkusil jsem z dokumentace první ukázku a ještě si vyžádal knihovnu `dplyr`. Proč si ji nenainstaloval sám?

```
> install.packages("dplyr")
> library(coronavirus)
> require(dplyr)
> coronavirus %>% filter(type=="confirmed") %>% group_by(Country.Region) %>%
    summarise(total=sum(cases)) %>% arrange(-total) %>% head(20)
# A tibble: 20 x 2
   Country.Region       total
   <chr>                <int>
 1 Mainland China       70446
 2 Others                 355
 3 Singapore               75
```

Tady jsem zbystřil. To jsou stará data v textové formě, nikoliv hezké mapy přes web s aktuálními daty. Ve slangové řeči: rtfm! To, co jsem právě nainstaloval, je knihovna https://github.com/RamiKrispin/coronavirus, která má stejný název. Mezi Issues jsem autorovi Trackeru zahlásil, že jeho název je v konfliktu s existující knihovnou z února 2020.

Pokračoval jsem v experimentování, knihovnu jsem odinstaloval a podíval se na návod, https://coronavirus.john-coene.com.

```
> remove.packages("coronavirus")
> install.packages("remotes")
> remotes::install_github("Johncoene/coronavirus")
```

Během instalace mi naskočila tato neobvyklá zpráva:

```
- Use ‘usethis::browse_github_pat()‘ to create a Personal Access Token.
- Use ‘usethis::edit_r_environ()‘ and add the token as ‘GITHUB_PAT‘.
```

Knihovna `usethis` se teprve instalovala. V každém případě zmínka o `Rate limit reset at: [...]`, kdy došlo k restartu za několik minut pro mne znamenalo chvíli počkat a instalaci zopakovat.

Před koncem instalace si R vyžádalo systémový balík `libpq-dev` ve starší verzi 10.3-1 (aktuální je 10.12-0). Udělal jsem hrubý krok, doporučuji čtenářům najít lepší řešení přes Docker či pečlivě projít, co se bude odinstalovávat.

```
$ sudo apt install aptitude
$ sudo aptitude install libpq-dev
```

Na první dotaz jsem dal nikoliv (`n`; starší balík by se nenainstaloval), na druhou nabídku ano (`y`). Zopakoval jsem instalaci v R a skončila úspěšně.

Zkusil jsem třířádkovou ukázku dle instalačního manuálu, coronavirus.john-coene.com (nutno zalistovat na webové stránce níž).

```
> library(coronavirus)
> virus<-crawl_coronavirus()
i Crawling data from John Hopkins
i Crawling data from Weixin
i Crawling data from DXY
> run_app(virus)
```

Pozn. Pokud bychom se dostali do konfliktu u příkazů, užijme:

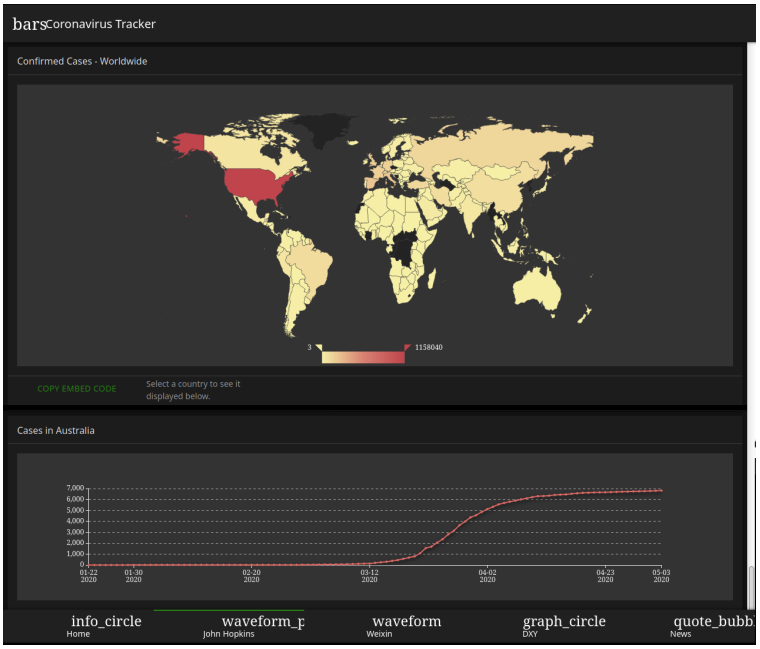
```
> virus <- coronavirus::crawl_coronavirus()
> coronavirus::run_app(virus)
```

Ve webovém prohlížeči se mi otevřela vygenerovaná stránka, pokaždé na jiném portu. Radost byla veliká! Za pozornost stojí, že má být Johns Hopkins, to již někdo zahlásil autorovi k opravení.

Ve spodní části je pět záložek. Na druhé (waveform_path) v bloku *China* a *World* a čtvrté (graph_circle) v bloku *Cities* jako kdyby něco chybělo. Po nakliknutí do bloku se otevře detailní výpis. Vrátit zpět se dá přes značku xmark_circle_fill v pravém horním rohu. Design je trochu nezvyklý, ale musíme mít na paměti, že je to zaměřené na mobilní telefony a já to zkoušel na notebooku.

Detaily kolem dat je možné nalézt v levém horním rohu pod bars nebo menu. Z R lze server zavřít přes klávesovou kombinaci Ctrl+C.

Nyní dokumentace radí si nastavit `crontab` atd. Co mne zaujalo u stažení dat z DXY je, že se občas nezadařilo připojit. V rychlosti jsem nahlédl na server https://education.rstudio.com/, konkrétně na dataio.



Jakmile se podařilo na servery připojit, mohl jsem si proměnnou `virus` uložit a opětovně užívat. Rychlá pomůcka u experimentů bez nutnosti aktualizace dat.

```
> save(virus, file="virus.RData")
> load("virus.RData")
```

4. Hašovací klíč od `newsapi.org` v2

Mou pozornost zaujala poslední záložka se zprávou `No newsapi token.` To bych rád poléčil. Autor v dokumentaci radí:

```
> library(coronavirus)
> create_config()
```

V pozadí se ze šablony vytvoří soubor `_coronavirus.yml`, blok `database` je povinný, blok `newsapi` volitelný. To byl pro mne problém. Já jsem to chtěl obráceně. Nevadí.

Přes `https://newsapi.org/register` jsem se zaregistroval a získal hašovací klíč. Zahlédl jsem jejich novou knihovnu pro R `newsanchor`, my zůstaneme u autorem užité knihovny `newsapi`.

Zkusil jsem R podsunout hašovací klíč:

```
> library(newsapi)
> newsapi::newsapi_key("41e22e9efcf64b2a9354a796b99c43b8")
```

Ale ani touto cestou ani jinou přes editaci souboru `_coronavirus.yml` se mi to nepodařilo.

Prvně jsem nahlédl na zdrojové kódy v:

```
$ cd ~/R/x86_64-pc-linux-gnu-library/4.0/coronavirus
```

Narazil jsem hlavně na binární soubory rds, rdx a rdb. Nejsem expert, abych dokázal odpovědět, jestli by se soubory daly rozluštit a editovat.

Prozkoumal jsem zdrojové kódy přímo od autora:

```
$ git clone https://github.com/JohnCoene/coronavirus
```

Došel jsem k závěru, že bych musel zdrojové kódy upravit, zkompilovat atd. To je nad rámec této sváteční zprávy.

V souboru `coronavirus/inst/app/Dockerfile` jsem si ověřil, že skutečně knihovnu `newsapi` přebírá z GitHubu od uživatele `news-r`.

5. PostgreSQL v10+190

Říkal jsem si, když už se mi podařilo nainstalovat `libpq-dev`, dokáži i zbytek. Otevřel jsem komunitní tutoriál. Nainstaloval jsem PostgreSQL:

```
$ sudo apt install postgresql postgresql-contrib
```

A nejkratší možnou cestou jsem se pustil do dalších kroků. Vytvořil jsem v databázovém systému nového uživatele `testing` a novou databázi `testing`. Vynechávám krok vytvoření uživatele pod operačním systémem.

```
$ sudo -i -u postgres createuser --interactive
Enter name of role to add: testing
Shall the new role be a superuser? (y/n) y
$ sudo -u postgres createdb testing
```

Uživatel je bez hesla, to webové rozhraní nepřijme. Nastavil jsem nové heslo přes:

```
$ sudo -i -u postgres
$ psql
postgres=# ALTER USER testing WITH PASSWORD 'testing';
postgres=# \q
$ exit
```

Rychlokurz `psql`: `help` je základní nápověda, `\l` je výpis databází, `\c testing` je připojení k naší databázi, `\dt` je výpis tabulek, `\h` je seznam

SQL příkazů, `\?` je seznam příkazů `psql` a `\q` ukončí běh programu. Verzálky u příkazů netřeba psát.

Vše zrealizované jsem zaznačil v `_coronavirus.yml`:

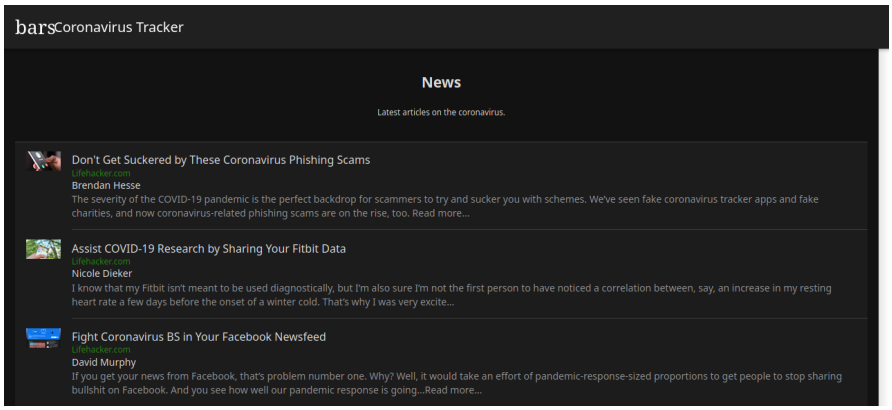
```
database:
  name: testing
  host: 127.0.0.1
  user: testing
  password: testing
newsapi:
  key: 41e22e9efcf64b2a9354a796b99c43b8
```

Když jsem opakoval tři řádky ukázkového spuštění v R, výpis se mi rozšířil o tyto dva řádky:

```
i Crawling news from newsapi.org
✔ Writing to database
```

V páté záložce mi vyběhly novinky, aktivovaný dotaz lze nalézt u autora v souboru `coronavirus/R/crawl.R`:

```
news <- newsapi::every_news("coronavirus OR covid", results = 100, language =
    "en", sort = "popularity")
```



Ověření funkčnosti můžeme zjistit i z tabulky `log`:

```
$ psql -h localhost -d testing -U testing
Password for user testing: testing
testing=# SELECT * FROM log;
```

Dostáváme přibližně takový výsledek:

```
         last_updated
-------------------------------
 2020-05-01 18:19:24.547171+02
(1 row)
```

Dle chuti lze dál bádat u surových dat, např.:

```
testing=# SELECT * FROM jhu WHERE country='Czechia';
testing=# SELECT * FROM jhu WHERE country='Slovakia';
testing=# \q
```

6. Bioconductor v3.11

Před dalším krokem si obvykle nastavuji plná práva u těchto adresářů:

```
$ cd /usr/lib/R
$ sudo chmod -R 777 site-library/
$ sudo chmod -R 777 library/
$ cd /usr/share
$ sudo chmod -R 777 R/
```

Za zmínku stojí, že manažer knihoven `biocLite` ustupuje a roli nahrazuje `BiocManager`. V R lze otestovat:

```
> install.packages("BiocManager")
> library(BiocManager)
> BiocManager::install()
> BiocManager::available()
```

Můžeme ověřit instalaci knihoven:

```
> BiocManager::valid()
   [...] "coronavirus", "echarts4r", "shinyMobile" [...]
Warning message:
0 packages out-of-date; 3 packages too new
```

To souhlasí, neb `coronavirus` byl instalován z GitHubu, nikoliv z CRANu.

Pro badatele stojí za pozornost obrazy pro Docker a Amazon (Amazon Machine Image, AMI). Je zde možnost instalovat vývojářské knihovny:

```
> BiocManager::install(version="devel")
```

7. Řešení konfliktu názvu knihovny v R

Na chvíli se ještě vraťme k řešení konfliktu stejného názvu knihoven. Na Stackoverflow zmiňují v principu dvě cesty.

Stáhnout si zdrojové soubory a nic neměnit:

```
$ cd /tmp
$ wget https://cran.r-project.org/src/contrib/coronavirus_0.1.0.tar.gz
$ R CMD INSTALL -l /tmp coronavirus_0.1.0.tar.gz
```

V R si pak volat jeden z příkazů a vybrat tak chtěnou knihovnu:

```
> # library(coronavirus)
> library(coronavirus, lib.loc="/tmp")
```

Když jsem zkoušel paralelně spustit i instalovanou knihovnu z GitHubu `coronavirus` odkomentováním prvního řádku, tak to neběželo. Tuším, že se jedná o bezpečnostní pojistku.

Druhá cesta je zasáhnout do souboru `DESCRIPTION`.

```
$ tar xvf coronavirus_0.1.0.tar.gz
$ mv coronavirus coronavirusRami
$ cd coronavirusRami
$ nano DESCRIPTION
```

První řádek upravit například na `Package: coronavirusRami`.

Volitelně upravíme i `MD5`, konkrétně první řádek za výpis:

```
$ md5sum -b DESCRIPTION
324f8275940bfa7fde376934c57a28ae *DESCRIPTION
```

Chtělo by to přejmenovat i další soubory na `coronavirusRami`, u této školní ukázky vynechávám. Zabalil jsem si zpět a už podsunul R:

```
$ cd ..
$ tar cvf coronavirusRami.tar.gz coronavirusRami/
$ R CMD INSTALL coronavirusRami.tar.gz
```

Ověřit funkčnost můžeme už přímo v R a lze si spustit oba konfliktní balíčky paralelně:

```
> library(coronavirus)
> ?coronavirus::crawl_coronavirus
> library(coronavirusRami)
> ?coronavirusRami::coronavirus
```

8. Pár tipů místo Závěru

Podobně jako jsou v R knihovny setříděné podle kategorií, viz Zobrazení úloh (CRAN Task Views), lze nahlédnout u RStudia na R Views s klíčovým slovem `covid-19`. K dnešnímu dni tam jsou tři záznamy: Some Select COVID-19 Modeling Resources, Simulating COVID-19 interventions with R a COVID-19 epidemiology with R.

Kdo by dal přednost odpočinku od R a PostgreSQL, nechť nahlédne na aktuální stavy kolem koronaviru na `https://www.twitch.tv/killars`.

TEXLIVE 2020: NOVINKY V TEXOVÉM SVĚTĚ
TEXLIVE 2020: NEWS IN THE WORLD OF TEX

Pavel Stříž

E-mail: pavel@striz.cz

Motto: *In the Beginning Was the Number*
Jean-Côme Charpentier @ TEXový balíček xlop

1. Instalace

Už mnoho let používám TEXLive na Xubuntu a snažím se každý rok o novou instalaci. Vše vyzkoušet a prozkoumat.

Z webové stránky http://tug.org/texlive/acquire-netinstall.html či přímo si stáhnu a rozbalím instalační skript do pracovního adresáře.

```
$ wget http://mirror.ctan.org/systems/texlive/tlnet/install-tl-unx.tar.gz
$ tar xvf install-tl-unx.tar.gz
$ cd install-tl-20200425 # změnit na aktuální časovou známku
$ ./install-tl
```

Obvykle nemám žádný problém a instaluji, u této verze se mi nepodařilo aktivovat `tlmgr update`, tak jsem si před instalací navolil adresář `~/texlive/2020`.

Po instalaci se rozšiřují či upravují systémové cesty (`MANPATH, INFOPATH` a především `PATH`), obvykle v souboru `~/.bashrc`. Po úpravě souboru si volám `source ~/.bashrc`, případně si otevřu nové terminálové okno.

Ověřujeme spustitelnost přes `which tex` nebo `tex --version`.

Aktualizace balíčků se realizuje přes `tlmgr update --self --all`.

Dokumentace balíčku se volá přes: `texdoc <balíček>`.

Je svátek 8. 5. 2020 a TEXLive 2020 mi nainstaloval 3999 balíčků.

2. Novinky

Není možné podchytit všechny novinky, ale přecejenom některé balíčky vyčnívají či by mohly pomoci.

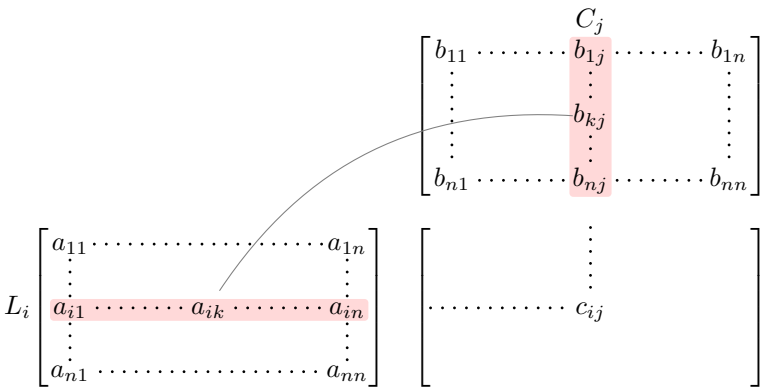
ConTEXt je samostatná kategorie, viz https://wiki.contextgarden.net, za LuaTEX sleduji https://ctan.org/topic/luatex a za LATEX3 pak https://ctan.org/topic/expl3. Nové zprávy na ctan.org lze sledovat na ctan-ann, témata jsou roztříděna na ctan.org/topics/highscore. Reálné TEXové problémy a odpovědi TEXistů hledejte na komunitním serveru https://tex.stackexchange.com/ (zkracováno TEX.SE).

2.1. Balíček nicematrix

Ti*k*Z vedle grafiky navrhl i sazbu tabulek a matic jako skládání pojmenovaných uzlů (angl. nodes). Balíček nicematrix rozšiřuje vizuální možnosti. Upravuje styl výpustků s možností se napojovat mezi uzly. Dokument se sází třikrát. Prvně se zjišťují rozměry pro knihovnu Ti*k*Zu fit a poté se vše správně umisťuje přes tikz. Autorem balíčku je François Pantigny.

```
$ texdoc nicematrix
```

```
%! lualatex-dev mal-nicematrix.tex
\documentclass{standalone}
\usepackage{tikz,nicematrix} \usetikzlibrary{fit}
\tikzset{highlight/.style={rectangle, fill=red!15, blend mode = multiply,
     rounded corners = 0.5 mm, inner sep=1pt, fit = #1}}
\begin{document}
\begin{NiceMatrixBlock}[auto-columns-width] \NiceMatrixOptions{nullify-dots}
$\begin{array}{cc}&
\begin{bNiceArray}{C>{\strut}CCCC}[name=B, first-row]
&& C_j\\ b_{11}  & \Cdots & b_{1j} & \Cdots & b_{1n} \\
\Vdots  && \Vdots && \Vdots \\ && b_{kj}\\
&& \Vdots\\b_{n1} & \Cdots & b_{nj} & \Cdots & b_{nn}\end{bNiceArray} \\ \\
\begin{bNiceArray}{CC>{\strut}CCC}[name=A, first-col]
& a_{11} & \Cdots &&& a_{1n} \\& \Vdots &&&& \Vdots \\
L_i & a_{i1} & \Cdots & a_{ik} & \Cdots & a_{in} \\
& \Vdots &&&& \Vdots \\ & a_{n1} & \Cdots &&& a_{nn} \\
\end{bNiceArray}& \begin{bNiceArray}{CC>{\strut}CCC}
& && \\ & & \Vdots \\ \Cdots & & c_{ij} \\ \\ \\
\end{bNiceArray} \end{array}$ \end{NiceMatrixBlock}
\begin{tikzpicture}[remember picture, overlay]
\node[highlight=(A-3-1)(A-3-5)]{}; \node[highlight=(B-1-3)(B-5-3)]{};
\draw[color=gray] (A-3-3) to [bend left] (B-3-3);
\end{tikzpicture}
\end{document}
```



2.2. Balíček witharrows

Od Pantignyho vznikl ještě jeden podobně zaměřený balíček. Tento je vhodný na popis úprav matematických vztahů. V dokumentaci je řada překlepů, chce to ještě vychytat.

```
$ texdoc witharrows
```

```
%! lualatex-dev mal-witharrows.tex
\documentclass{article} % se standalone balíček zlobil
\pagestyle{empty}
\usepackage{witharrows}
\begin{document}
\def\ee{\mathrm{e}} \def\ii{\mathrm{i}}
\begin{DispWithArrows*}[displaystyle, wrap-lines]
S_n & = \frac1n \Re
    \left(\sum_{k=0}^{n-1}\bigl(\ee^{\ii\frac{\pi}{2n}}\bigr)^k\right)
\Arrow{\upshape sum of terms of a geo\-metric progression of ratio
    $\ee^{\ii\frac{2\pi}n}$}\\
& = \frac1n \Re \left( \frac{1-\bigl(\ee^{\ii\frac{\pi}{2n}}\bigr)^n}
{1-\ee^{\ii\frac{\pi}{2n}}}\right)
\Arrow{\upshape This line has been wrapped automatically.} \\
& = \frac1n \Re \left(\frac{1-\ii}{1-\ee^{\ii\frac{\pi}{2n}}}\right)
\end{DispWithArrows*}
\end{document}
```

$$
\begin{aligned}
S_n &= \frac{1}{n}\Re\left(\sum_{k=0}^{n-1}\left(\mathrm{e}^{\mathrm{i}\frac{\pi}{2n}}\right)^k\right) && \text{sum of terms of a geometric progression of ratio } \mathrm{e}^{\mathrm{i}\frac{2\pi}{n}} \\
&= \frac{1}{n}\Re\left(\frac{1-\left(\mathrm{e}^{\mathrm{i}\frac{\pi}{2n}}\right)^n}{1-\mathrm{e}^{\mathrm{i}\frac{\pi}{2n}}}\right) && \\
&= \frac{1}{n}\Re\left(\frac{1-\mathrm{i}}{1-\mathrm{e}^{\mathrm{i}\frac{\pi}{2n}}}\right) && \text{This line has been wrapped automatically.}
\end{aligned}
$$

2.3. Balíček siunitx

Přes balíček nicematrix jsem narazil na balíček siunitx od Josepha Wrighta. Užíval jsem balíčky siunits a pgfplotstable, tohle je pravděpodobný nástupce na sazbu jednotek a tabulek s čísly.

```
$ texdoc siunitx
```

```
%! lualatex-dev mal-siunitx.tex
\documentclass[varwidth]{standalone}
\usepackage{cancel}
\usepackage[binary-units]{siunitx}
\DeclareSIPrePower\quartic{4}
\DeclareSIPostPower\tothefourth{4}
```

```
\begin{document}
\num{1+-2i}, \num{.3e45}, \si{kg.m.s^{-1}},
\si{\kilogram\metre\per\second}, \si{\joule\per\mole\per\kelvin},
\si[per-mode=fraction]{\cancel\kilogram\metre\per\cancel\kilogram\per\second},
\si{\kilogram\tothefourth}, \si{\quartic\metre}, \par \num{1e2/3e4},
\ang{6;7;6.5}, \ang[angle-symbol-over-decimal]{45.697},
\SI{100}{\mebi\byte}, \SI[prefixes-as-symbols=false]{30}{\kibi\bit},
\SI{1.234}{\metre}, \SI[locale = DE]{6.789}{\metre}
\end{document}
```

$1 \pm 2\mathrm{i}$, 0.3×10^{45}, $\mathrm{kg\,m\,s^{-1}}$, $\mathrm{kg\,m\,s^{-1}}$, $\mathrm{J\,mol^{-1}\,K^{-1}}$, $\frac{\cancel{\mathrm{kg}}\,\mathrm{m}}{\cancel{\mathrm{kg}}\,\mathrm{s}}$, kg^4, m^4,
$1 \times 10^2/(3 \times 10^4)$, 6°7′6.5″, 45.697, 100 MiB, 30×2^{10} bit, 1.234 m, 6,789 m

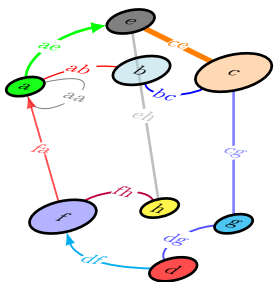
2.4. Balíček tikz-network

Na sazbu obrázků z teorie grafů existuje nespočet nástrojů, např. tkz-graph. U tohoto balíčku od Jürgena Hackla mne zaujaly vrstvy ve 3D. V pracovním adresáři jsem si nalinkoval pomocné soubory a ukázky se rozběhly.

```
$ mkdir data
$ cd data
$ ln -s <cesta>/texmf-dist/doc/latex/tikz-network/data/ml_{vertic,edg}es.csv .
$ cd ..

$ texdoc tikz-network

%! lualatex-dev mal-tikz-network.tex
\documentclass{standalone}
\usepackage{tikz-network}
\begin{document}
\begin{tikzpicture}[multilayer=3d]
\Vertices{data/ml_vertices.csv}
\Edges{data/ml_edges.csv}
\end{tikzpicture}
\end{document}
```



2.5. Balíček xlop

Autorem je Jean-Côme Charpentier. Balíček nám pomáhá se sazbou základních aritmetických operací a schémat. Zdeněk Wagner mi psal, že autor postrádá v dokumentaci informaci, že schéma pro násobení, které se stále učíme na základních školách, vytvořil někdy v 8. století podle indických knih perský matematik ابو جعفر الخوارزمی ابن موسی محمد ابو عبد الله, krátce Al-Chorezmí. Autorovi jsem postřeh napsal. Na pomoc s arabštinou jsem si vzal balíček arabluatex od Roberta Alessiho s renovovaným písmem Amiri od Khaleda Hosnyho.

Historická vsuvka. Díky překladům Al-Chorezmího spisů se seznamujeme s algebrou, číslem nula a nejspíše i s x pro neznámou psáno tehdy jako **X** (arabsky aš-šáí, doslova věc). Ve středověku bylo jméno Al-Chórezmí latinizované na Al-Gorizmí, které bylo základem slova algoritmus.

```
$ texdoc xlop
```

```
%! lualatex-dev mal-xlop.tex
\documentclass[varwidth,border={0 0 0 2pt}]{standalone}
\usepackage{xlop}
\begin{document}
\opdiv[style=text]{124}{7},\qquad\opidiv[style=text]{124}{7},\qquad
    \opdiv[period,style=text,equalsymbol=$\approx$,hrulewidth=0.5pt,
    vruleperiod=0.7]{150}{7}\par\medskip\hfil\opadd{012.3427}{5.2773}
\qquad\opmul[displayshiftintermediary=all]{453}{1001205}
\end{document}
```

$124 \div 7 \approx 17.71428571, \qquad 124 = 7 \times 17 + 5, \qquad 150 \div 7 \approx 21.\overline{428571}\ldots$

$$\begin{array}{rr} & {\scriptstyle 1\,1\,1} \\ & 1\,2.3\,4\,2\,7 \\ + & 5.2\,7\,7\,3 \\ \hline & 1\,7.6\,2 \end{array} \qquad \begin{array}{rr} & 4\,5\,3 \\ \times & 1\,0\,0\,1\,2\,0\,5 \\ \hline & 2\,2\,6\,5 \\ & 9\,0\,6\,\cdot\,\cdot \\ & 4\,5\,3\,\cdot\,\cdot\,\cdot \\ 4\,5\,3\,\cdot & \cdot\,\cdot\,\cdot\,\cdot \\ \hline 4\,5\,3\,5\,4\,5\,8\,6\,5 & \end{array}$$

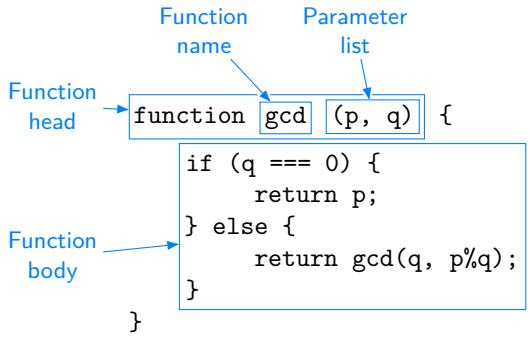
2.6. Balíček codeanatomy

Již v dobách ranných bylo možné najít typografické vychytávky na sazbu algoritmů, zdrojových kódů a pseudokódů. Tento balíček zvýrazňuje části kódu s možností je popsat. TEXujeme dvakrát. Autorem je Hồng-Phúc Bùi.

```
$ texdoc codeanatomy
```

```
%! lualatex-dev mal-codeanatomy.tex
\documentclass{standalone}
\usepackage{codeanatomy}
```

```
\begin{document}
\begin{tikzpicture}[remember picture]
\codeBlock{%
\cPart{functionHead} {function \cPart{functionName}{gcd}
    \cPart{paramList}{(p, q)}} \{ \\[2.5pt]
\ptab{}\mtPoint{mostLeft}if (q === 0) \{ \\
\ptab\ptab{} return p; \\
\ptab\} else \{ \\
\ptab\ptab{} return gcd(q, p\%q);\extremPoint{mostRight} \\
\ptab\mbPoint{mostBottom}\} \\
\} }% end od \codeBlock
\fitExtrem{functionBody}{(mostLeft) (mostRight) (mostBottom)}
\codeAnnotation{functionHeadText}(-1,3){Function\\head}
\codeAnnotation{functionBodyText}(-1,1){Function\\body}
\codeAnnotation{functionNameText}( 1,4){Function\\name}
\codeAnnotation{paramListText} ( 3,4){Parameter\\list}
\draw[->,annotation] (functionHeadText)  -- (functionHead);
\draw[->,annotation] (functionBodyText)  -- (functionBody);
\draw[->,annotation] (functionNameText)  -- (functionName);
\draw[->,annotation] (paramListText) --  (paramList);
\end{tikzpicture}
\end{document}
```



2.7. Balíček mercatormap

V roce 2018 na konferenci OSSConf v Žilině Aleš Kozubík představil z pohledu uživatele balíček getmap. Ten pracuje s OpenStreetMap. Tehdy to byl nový balíček i pro mne a příjemné překvapení. Letos jsem organizátory předběhl, protože jsem jako první objevil tento balíček. Je to cenné hlavně z pohledu propojení dvou sekcí: TEXové a GISácké. Autorem je Thomas F. Sturm.

Je potřeba mít Python3 a několik balíčků, v mém případě to bylo:

```
$ sudo apt install python3
$ sudo -H pip3 install Pillow requests
```

Přes tento balíček se mi nepodařilo získat mapy z mapy.cz ani ze serveru freemap.sk. V pozadí se očekává na dotaz png soubor, obdrží html. Český server sice API má na api.mapy.cz, ale nikoliv s touto možností. Slovenský server také umí, ale k png se musí člověk proklikat v rámci exportu mapy. Napsal jsem to vývojářům jako tip na rozšíření, kdyby se náhodou nudili, neb minimálně Sturm v dokumentaci píše, že rád nový mapový server do dokumentace svého balíčku zařadí.

Přikládám mapovou ukázku, v poznámkách v kódu je nefunkční část rozhraní na `mapy.cz`, na slovenský server by to bylo obdobné, to pro případ, že by to v budoucnu fungovalo. Je potřeba TEXovat s parametrem `--shell-escape` (Unix), případně `--enable-write18` (Microsoft Windows). Za běhu se dočasné soubory ukládají do složek `maps` a `tiles`.

```
$ texdoc mercatormap

%! lualatex-dev --shell-escape mal-mercatormap.tex
\documentclass{standalone}
\usepackage{mercatormap}
\mermapset{python=python3}
\mrcactivatescript
\begin{document}
\begin{tikzpicture}
\mrcmap[type=reference, latitude=49.14549, longitude=16.99913, flex reference
     scale=250000, source=topplusopen p250, target=wmsmap, tex
     width=0.9\linewidth, tex height=3cm]{mapa-bucovice}
\mrcdrawmap
\node[below, font=\bfseries\sffamily] at (mrcmap.south) {Bučovice, rodiště
     autora zprávy};
\end{tikzpicture}
%\mrcnewsupplysource{mapycz}{
%  url={https://en.mapy.cz/zakladni?x={x}&y={y}&z={z}},
%  attribution={mapycz},
%  attribution print={mapycz},
%  basename=tiles/mapycz}
\end{document}
```

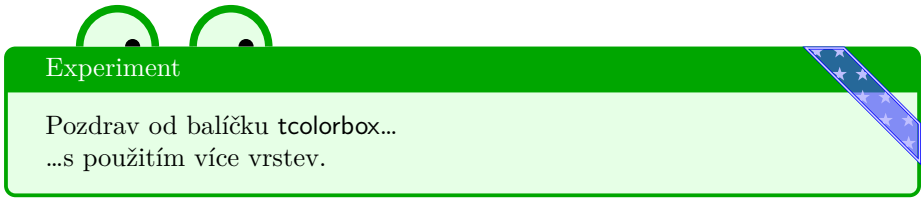
Bučovice, rodiště autora zprávy

2.8. Balíček tcolorbox

Storm v dokumentaci balíčku mercatormap masivně používá tento balíček, jehož je i autorem. Ačkoliv balíček znám a je vhodný především do prezentací, různých poznámek do knih a skript, na plakátky a obálky knih, přecejenom na něm autor dále pracuje a stojí za připomenutí. Já jsem si z obří dokumentace čítající přes 500 stran vytáhl žabího prince, nu, spíš obyčejnou žabu.

```
$ texdoc tcolorbox
```

```
%! lualatex-dev mal-tcolorbox.tex
\documentclass{standalone}
\usepackage{tikz}
\usepackage{tcolorbox}
\usetikzlibrary{patterns}
\tcbuselibrary{skins,hooks}
\tcbset{frogbox/.style={enhanced, colback=green!10, colframe=green!65!black,
    enlarge top by=5.5mm, overlay={\foreach \x in {2cm,3.5cm} {
\begin{scope}[shift={([xshift=\x]frame.north west)}]
\path[draw=green!65!black, fill=green!10, line width=1mm] (0,0) arc
    (0:180:5mm);
\path[fill=black] (-0.2,0) arc (0:180:1mm);
\end{scope}}}]}}
\tcbset{ribbon/.style={overlay app={\path[fill=blue!75!white, draw=blue,
    double=white!85!blue, preaction={opacity=0.6, fill=blue!75!white}, line
    width=0.1mm, double distance=0.2mm, pattern=fivepointed stars, pattern
    color=white!75!blue] ([xshift=-0.2mm, yshift=-1.02cm]frame.north east)
    -- ++(-1,1) -- ++(-0.5,0) -- ++(1.5,-1.5) -- cycle;}}}
\begin{document}
\begin{tcolorbox}[frogbox, ribbon, title=Experiment]
Pozdrav od balíčku \textsf{tcolorbox}\ldots\par\ldots s použitím více vrstev.
\end{tcolorbox}
\end{document}
```



2.9. Balíček tikz-planets

S úsměvem píši, že s příchodem Lua (z portugalštiny měsíc) se hodí takový balíček. Zde je ukázka vysázení fází Měsíce.

```
$ texdoc tikz-planets
```

```
%! lualatex-dev mal-tikz-planets.tex
```

```
\documentclass{standalone}
\usepackage{planets}
\begin{document}
\begin{tikzpicture}[scale=0.7]
\planet[surface=moon, phase=new, centerx=0]
  \node at (0, 0) {new};
\planet[surface=moon, phase=first crescent, centerx=2]
  \node[align=center] at (2, 0) {first \\ crescent};
\planet[surface=moon, phase=first half, centerx=4]
  \node[align=center] at (4, 0) {first \\ half};
\planet[surface=moon, phase=waxing gibbous, centerx=6]
  \node[align=center] at (6, 0) {waxing \\ gibbous};
\planet[surface=moon, phase=full, centerx=8]
  \node[align=center] at (8, 0) {full};
\planet[surface=moon, phase=waning gibbous, centerx=10]
  \node[align=center] at (10, 0) {waning \\ gibbous};
\planet[surface=moon, phase=last half, centerx=12]
  \node[align=center] at (12, 0) {last \\ half};
\planet[surface=moon, phase=last crescent, centerx=14]
  \node[align=center] at (14, 0) {last \\ crescent};
\end{tikzpicture}
\end{document}
```

2.10. Balíček emoji

Program HarfBuzz (GitHub) umí vykreslovat písma jako třeba známější program Pango. První pokusy o zařazení do TEXu jsem viděl u Michala Hofticha, novější je pokus u Lua modulu od Deepaka Joiseho. Další testy lze nalézt v článcích v TUGboatu od Khaleda Hosnyho (GitHub) či v MAPS od Kaie Eignera (GitHub). Pro nás smrtelníky se jedná o užití barevných a exotických písem. Do LuaTEXu knihovnu zařadil Luigi Scarso a tým LuaTEXu.

V PlainTEXu se užívá `luahbtex` a v LATEXu `lualatex-dev`. To bylo nutné ještě v TEXLive 2019. Od TEXLive 2020 stačí opět užívat `lualatex`. LATEXový formát jsem užíval u všech zmíněných ukázek této zprávy.

Zde vstupuje do popředí balíček emoji od Xiangdong Zeng (曾祥东). Na některá písma mě navedla dokumentace, některá jsem si stáhl. První a poslední písmo je rastrové, zbytek jsou písma vektorová. Druhé písmo není v barvě. V balíčku je předvolené první písmo. Pokus o rozšíření citací o emoji zkusil Leon Sixt v úsměvném balíčku emojicite.

```
$ wget -O EmojiOneMozilla.ttf https://github.com/mozilla/positron/blob/master/
    browser/fonts/EmojiOneMozilla.ttf?raw=true
$ wget -O AppleColorEmoji.ttf https://github.com/potyt/fonts/blob/master/
    macfonts/Apple%20Color%20Emoji/Apple%20Color%20Emoji.ttf?raw=true
```

```
$ texdoc emoji
```

```
%! lualatex-dev mal-emoji.tex
\documentclass{standalone}
\usepackage{emoji}
\begin{document} \fontsize{19}{19}\selectfont
\def\mallist#1{%
\setemojifont{NotoColorEmoji.ttf}\emoji{#1}% Předvolené
\setemojifont{NotoEmoji-Regular.ttf}\emoji{#1}%
\setemojifont{EmojiOneMozilla.ttf}\emoji{#1}%
\setemojifont{TwemojiMozilla.ttf}\emoji{#1}%
\setemojifont{AppleColorEmoji.ttf}\emoji{#1}}%
\mallist{joy} \mallist{kiss-mark} \mallist{+1}
\end{document}
```

2.11. Balíček pgfornament

Na odlehčenou zmíním ještě jeden balíček, který je přepracován přes Ti*k*Z a dává tak možnost zasáhnout do různých kreseb a udělat z nich malbu. Autorem je Alain Matthes, ornament vlevo na další straně. Velkou inspirací ke vzniku byl balíček pgfornament-han z roku 2018 od LianTze Lim (林莲枝) a Chennan Zhang (张晨南), viz ornament vpravo.

```
$ texdoc pgfornament
```

```
%! lualatex-dev mal-pgfornament.tex
\documentclass[margin=1pt]{standalone}
\usepackage[dvipsnames]{xcolor}
\usepackage{pgfornament}
\usepackage{pgfornament-han}
\begin{document}
\tikzset{pgfornamentstyle/.style={draw=green!20!black, fill=orange, fill
    opacity=.5, scale=0.7, ultra thick}}%
\tikz\node{\pgfornament{3}};
\tikzset{pgfornamentstyle/.style={draw=Goldenrod, fill=Red, line width=1pt}}
\tikz\node[fill=black, circle, draw=Red, line width=2pt, inner
    sep=-8pt]{\pgfornamenthan[scale=0.25]{56}};
\end{document}
```

2.12. Za pozornost ještě stojí

Již bez ukázek upozorňuji na další nástroje a balíčky.

- xindex od Herberta Voße je na LuaTEXu založený rejstříkový procesor. Je to aktivní vývojář, hlavně kolem projektu PSTricks a autor mnoha knih a dokumentace balíčků.
- tex4ebook je na LuaTEXu založený balíček na převod z LATEXu do elektronické knihy od českého vývojáře Michala Hofticha.
- lwarp je podobně smýšlející projekt na převod z LATEXu do HTML5 od Briana Dunny.
- Nelze zapomenout na neustále vylepšovaný obří nástroj na přípravu seznamu literatury biblatex s jeho balíčky.
- V neposlední řadě balíček ocgx2, který je nástupcem balíčků ocgx a ocg-p od Alexandra Grahna, mj. autora balíčků media9, animate a nového experimentálního balíčku media4svg.
- O nástroji dvisvgm, který se užívá v pozadí balíčku media4svg či nástroje Manim na matematické animace, ještě uslyšíme, protože plánují vedle převodu z dvi do svg i převod pdf do svg.

3. METAPOST ztracen a nalezen

METAPOST nahradil METAFONT na kresbu. Pamatuji si své začátky nad příklady Vincenta Zoonekynda (archiv). Dnes je METAPOST integrován do ConTEXtu přímo jako knihovna, zájemce odkazuji na ConTEXt Garden.

Jaromír Antoch se mne ptal, jestli by dokázal dostat vektorovou podobu svých kreseb na webové stránky. Když opomineme rastr, formát pdf samotný či konverzi do jiných formátů, tak stojí za pokus to vyzkoušet přímo v METAPOSTu. V minulých letech se totiž do zásahů pustil Taco Hoekwater, jeden z jeho nápadů byl rozšířit výstup do svg.

Vzal jsem si do parády ukázku č. 32 od Vincenta Zoonekynda, upravil jsem ji dle návodu v dokumentaci `texdoc metapost`, str. 5, do následující podoby. Jen jsem v proměnné `outputtemplate` místo `mps` užil `svg`:

```
%! mpost zoonek.mp
prologues := 3;
outputtemplate := "%j-%c.svg";
outputformat := "svg";
beginfig(32)
  u:=1cm; pair A,B,C,D,E,F,G;
  A := (-u,u); B := (0,u); C := (u,u); D := (-u,0); E := (0,0); F := (u,0);
  draw A--D; draw A--E; draw A--F;
  draw B--D; draw B--E; draw B--F;
  draw C--D; draw C--E; draw C--F;
  dotlabel.top(btex $a$ etex, A); dotlabel.top(btex $b$ etex, B);
  dotlabel.top(btex $c$ etex, C); dotlabel.bot(btex $a'$ etex, D);
  dotlabel.bot(btex $b'$ etex, E); dotlabel.bot(btex $c'$ etex, F);
endfig;
bye.
```

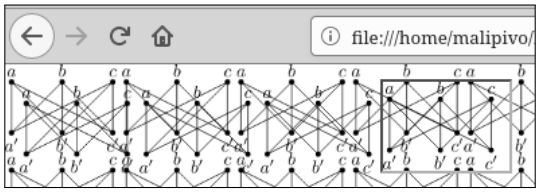
Spustil jsem poté:

```
$ mpost zoonek.mp
$ inkscape zoonek-32.svg &
$ firefox zoonek-32.svg &
```

První řádek vygeneruje soubor `zoonek-32.svg`, druhý řádek soubor otevře pro případnou úpravu a poslední řádek otevře soubor přímo v prohlížeči.

Za pomoci webové nápovědy jsem zkusil vložit obrázek do webové stránky `index.htm` a tu si pak přes `firefox index.htm` otevřít. Jedná se o čtyři základní způsoby vložení svg plus pátou cestu přes kaskádový styl CSS jako opakující se obrázek v pozadí. Snad se v náhledu zorientujete. Určitě existuje nespočet dalších způsobů, nechávám hlubší bádání na čtenáři.

```
<!DOCTYPE html>
<html>
<head><meta charset="UTF-8"/>
  <style>body{background-image: url(zoonek-32.svg);}</style>
</head>
<body>
  <img src="zoonek-32.svg" />
  <object type="image/svg+xml" data="zoonek-32.svg"></object>
  <embed type="image/svg+xml" src="zoonek-32.svg" />
  <iframe src="zoonek-32.svg" width="70px" height="50px"></iframe>
</body>
</html>
```



„Mně se to líbilo a potvrdilo mi to, že stojí, jde-li to, dělat věci nad základem, který bývá stálý, zatímco balíčky vymírají se svými tvůrci…“

Jaromír Antoch

Nesmrtelná slova. Vykreslit je do kamene!

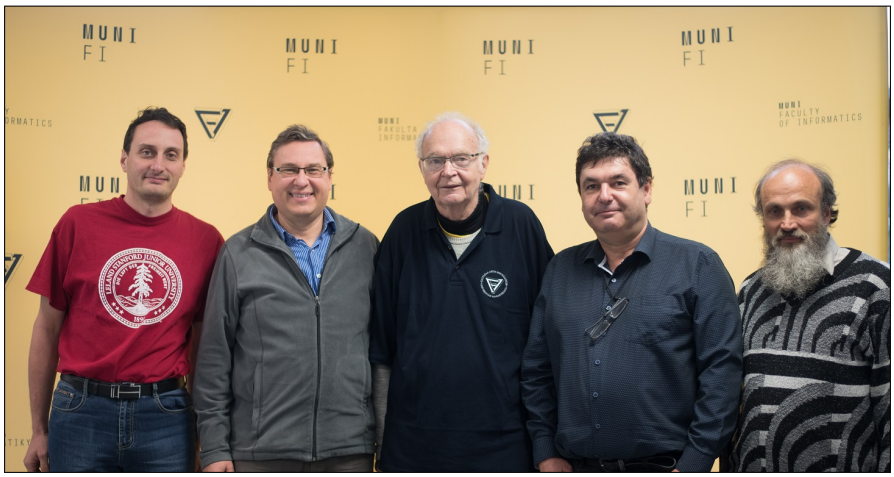
4. Co dodat závěrem?

Tohle vše máme opět k dispozici zadarmo, se zdrojovými kódy na přípravu čehokoliv a na dosah klávesnice.

Jo, abych nezapomněl: Donald E. Knuth alias DEK alias 高德纳 byl v září 2019 v Brně na Fakultě informatiky Masarykovy univerzity u příležitosti 25. výročí založení fakulty. Je tam přednáška, fotky ad.

www.fi.muni.cz/events/2019-celebrations-of-25-years-of-fi.html

V tu dobu jsem ve stavech zoufalství sázel cosi v jakémsi TEXu, tak jsem přednášku, diskuze a varhanní koncert vynechal. Možná by mi „Grand Wizard“ poradil. Kdoví!



Zleva: Jan Šustek, Jiří Rybička, DEK, Petr Sojka a Tomáš Hála

ROBUST 2020: NOVINKY
ROBUST 2020: NEWS

Redakce

BARDĚJOV
6. – 11. června 2021

SOBOTA 25. DUBNA 2020 : DEFINITIVNÍ POSUN

Jakkoliv se světlo na konci tunelu pomalu blíží, nadcházející chaos rozvolňování neřízeně řízený našimi soudy má pro nás nejasné dopady. Po diskusi s řadou z Vás jsme se nakonec rozhodli ROBUST 2020 přenést o rok na prakticky to samé datum, tj. **6. – 11. června 2021** na tom samém místě a v těch samých prostorách.

Hlavním důvodem nejsou ani tak nejasnosti kolem cestování, to se do podzimu alespoň v rámci Česko-Slovenska nějak usadí, ale především nejasnosti týkající se výuky na podzim, dobíhajícímu zkoušení, začátku zimního semestru na jednotlivých fakultách, atd.

Všem, kteří zaplatili vložné, je postupně vracíme, neboť jej budou muset školám i grantovkám vrátit. Žádáme Vás nicméně, aby si akci naplánovali na rok 2021. Podobně jako olympiáda doufáme i my napřesrok v nejméně tak hojnou účast, jaká se chystala letos. Díky za podporu a především, brzy na viděnou v lepších a klidnějších časech.

ÚTERÝ 24. března 2020 : COVID-19

Vzhledem k množícím se dotazům týkajícím se dopadu pandemie COVID-19 na Robust 2020 bychom chtěli říci, že se stále více obáváme, že jej budeme muset přeložit, podobně jako MOV dnes přeložil olympijské hry. **Na kdy? To teď nevíme a nechceme dělat předčasné rozhodnutí, které bychom museli opakovaně měnit.** Jakmile bude jasněji, samozřejmě se ozveme. V této chvíli, na rozdíl od některých médií a řady pseudo expertů, nevidíme světlo na konci tunelu. Ono tam samozřejmě je, ale zatím až někde za zatáčkou. Takže se prosím i nadále těšte, ROBUST 2020 se určitě uskuteční!!!

S pozdravem ROBUSTu 2020 zdar JA + GD + DH + IS

RSJ

PRVNÍ KONTAKT S PROGRAMEM MANIM
THE FIRST ENCOUNTER WITH MANIM

Pavel Stříž

E-mail: pavel@striz.cz

Abstrakt: Tento článek tvoří úvod do práce s programem Manim (matematické animace) založeném na Pythonu s podporou TEXu. Jedná se především o rešerši zdrojů, tipy, triky a řešení některých problematických partií. Program vytváří a spravuje Grant Sanderson alias 3blue1brown a Ben Eater. Původně to byl soukromý projekt pomáhající jim programově vytvořit náročnější videa, nyní se jedná o otevřený software.

Klíčová slova: Animace, Manim, Python, TEX.

Abstract: This article is an introduction to work with Manim software (Mathematical Animation Engine), which is a Python-based program with support of TEX. The paper consists mainly of research of sources, tips, tricks and solution to some problematic parts. Software is being developed and maintained by Grant Sanderson alias 3blue1brown and Ben Eater. It was initially a private project supporting them programatically create complex videos. Now, it's an open-source software.

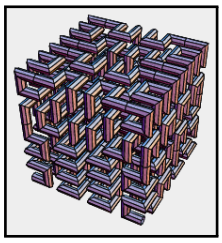
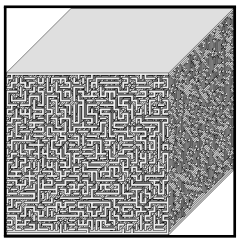
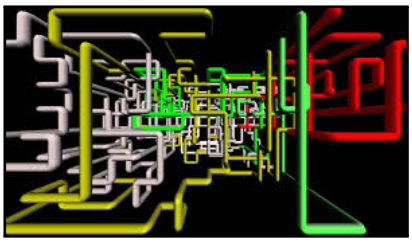
Keywords: Animation, Manim, Python, TEX.

Motto: Python se má rád, je to tam jedno `selfí` za druhým!

1. Mohou za to kvaterniony

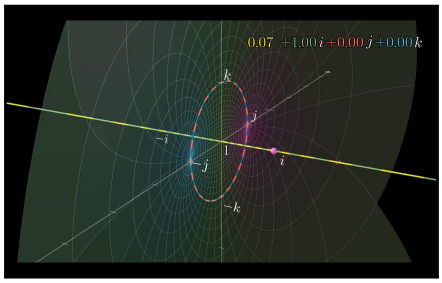
Někdy v roce 2018 jsem otevřel jeden ze starých problémů, a to jak vykreslit nekonečnou trubku, ale tak, aby se vzájemně nekřížila. Je to trochu obdoba Hilbertovy křivky (vlevo) či tvorba bludišť, např. v programu Daedalus (uprostřed). Zájemce odkazuji na demo (Xubuntu 18.04, obrázek vpravo):

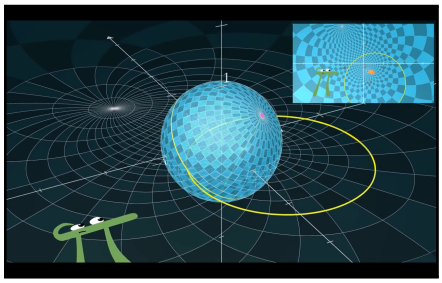
```
$ sudo apt install xscreensaver xscreensaver-gl
$ xscreensaver &
$ sleep 8; xscreensaver-command -demo 39
```

Podařilo se mi to vyřešit v Blenderu, ale tehdy dokumentace ke kvaternionům (rozšíření komplexních čísel pro 3D) byla strohá, hledal jsem doplňující materiály. Zaujalo mě video youtube.com/watch?v=d4EgbgTm0Bg s detaily na eater.net/quaternions, ale co víc, na fóru byla zmínka, že video bylo vytvořené v programu Manim. Tak jsem se do toho víc ponořil, neb mi název programu nic neříkal.

Zde je pár ukázek od tvůrců ze zmíněného videa:



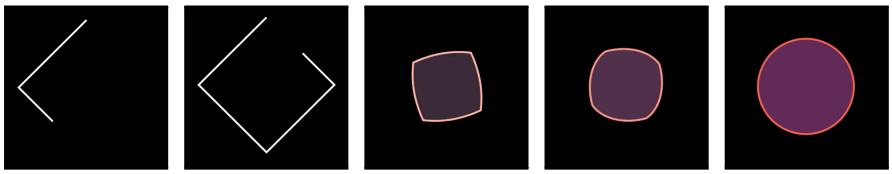


2. Hello World! aneb Manim animuje čtverec na kolečko

Jedná se o vyvíjený program na matematické animace a je docela programátorský oříšek dát vše do latě (instalaci, rozběhnutí ukázek, vlastní tvorba). Bral jsem to však za součást učení se.

Program lze nainstalovat přes `pip3` (manimlib), `virtualenv`, `docker`, `anaconda`, ale hlavně přímo. Detaily jsou na github.com/3b1b/manim.

```
$ git clone https://github.com/3b1b/manim.git
$ cd manim
$ sudo -H pip3 install -r requirements.txt
$ python3 manim.py example_scenes.py SquareToCircle -pl
```



Pokud vše proběhne v pořádku spustí se vám video, kdy se bílý čtverec změní na barevné kolečko a to pak zmizí.

Problém jsem měl s instalací knihovny pycairo, to jsem vůči manuálu vyřešil přes:

```
$ sudo apt install python3-cairo
$ sudo -H pip3 install manimlib --ignore-installed pycairo
```

Druhý řádek mi v roce 2020 nejel, stále se mi snažil vnutit `pycairo`, a to při instalaci spadne, tak jsem užil první řádek z předchozího skriptu, v `requirements.txt` jsem zakomentoval řádek s `pycairo` a doinstaloval jen další závislosti:

```
$ sudo -H pip3 install -r requirements.txt
```

Touto cestou užívám starší pycairo verze 1.16.2 a Manim z pracovního adresáře, vše dále představené běží.

3. Hlouběji u spuštění z příkazového řádku

Nápovědu lze získat přes `python3 -m manim --help`. Ukázky jsou realizovány přes třídy, jejich seznam jsem si nechal vypsat přes `cat <soubor.py> | grep <třída či class>`. Obvykle vidíme `Scene`, `GraphScene`, `ThreeDScene` a `SVGMobject`.

Vytvoření videa se realizuje přes:

```
python3 -m manim <soubor.py> [třídy] [parametry]
```

Třídy oddělujeme mezerami. Bez zadání třídy vyběhne nabídka a zadáváme čísla oddělená čárkou. Ne všechny třídy jsou takto spustitelné.

Nejběžnější volitelné parametry jsou následující:

- `-p` = preview, otevření souboru po dokončení,
- `-l` | `-m` | bez parametru = velikost videa, low | medium | high,
- `-t` = transparent, pozadí bude průhledné,
- `-c` = color, pozadí bude mít specifickou barvu,
- `-r` = resolution, rozlišení v pixelech, výška čárka délka,
- `-s` = save the last frame, uloží z videa jen poslední snímek,
- `--livestream` | `--to-twitch` = živé vysílání,
- `-h` | `--help` = nápověda.

Výsledky se ukládají do adresáře `media/video`, pak do složky dle názvu skriptu, následuje složka `výška p rychlost`, např. `480p15` znamená 480 pixelů je výška obrazu v rychlosti 15 snímků/vteřinu. Vzniká série malých videí, které se v závěru spojí do velkého souboru mp4.

Generované TEXové soubory, xdv a svg lze nalézt v adresáři `media/TeX`.

4. Inspirativní zdroje

Mezi tutoriály v angličtině řadíme talkingphysics.wordpress.com/2019/01/08/getting-started-animating-with-manim-and-python-3-7 s pod-

půrnými soubory na github.com/zimmermant/manim_tutorial. Velkou inspirací jsou i zdrojové kódy na github.com/Solara570/demo-solara. Další tutoriál lze nalézt na github.com/malhotra5/Manim-Tutorial, obsáhlý v čínštině pak na github.com/cai-hust/manim-tutorial-CN.

Kdo dává přednost videotutoriálům, nechť nahlédne na youtube.com/channel/UCxiWCEdx7aY88bSEUgLOC6A s podpůrnými soubory na github.com/Elteoremadebeethoven/AnimationsWithManim.

Za poklad ke zkoumání však lze považovat přímo ve složce `manim` adresář `from_3b1b`, speciálně adresář `old`. Řada kódů však nejede na první dobrou, neb autoři Manim přepracovávají a upravují. Nechávám otevřené pro badatele, je to spleť překlepů, programátorských háků a háčků v Pythonu.

V češtině v akad. roce 2017/2018 proběhl seminář od Mirka Olšáka, http://www.olsak.net/mirek/manim/, od března 2020, možná v souvislosti s koronavirem, vznikají překlady anglických videí, www.youtube.com/channel/UCIhWS2rX78OXidZ87QLkkxA.

5. Otázky a odpovědi

Představím některé běžné situace a jejich řešení.

5.1. Jak řešit diakritické znaky?

Jinými slovy, lze zasáhnout do preambule TEXové šablony? Šablona je uložena v souboru `manimlib/tex_template.tex`, po záloze souboru jsem jej upravil do této podoby:

```
\documentclass[preview]{standalone}
\usepackage[czech]{babel}
\usepackage[utf8]{luainputenc}
\usepackage[IL2]{fontenc}
\usepackage{amsmath,amssymb}
\usepackage{halloweenmath,tikz,fdsymbol}
\renewcommand\rmdefault{cmr}
\renewcommand\sfdefault{cmss}
\renewcommand\ttdefault{cmtt}
\begin{document}
YourTextHere
\end{document}
```

V pozadí se užívá `latex` s převodem do svg. Přepnul jsem si na `lualatex` s jedním parametrem navíc v `manimlib/utils/tex_file_writing.py`:

```
commands = [
    "lualatex",           # byl "latex",
    "-output-format=dvi", # přidaný řádek
```

Náš kód v souboru `ukazky/ahoj-svete.py` by mohl vypadat takto:

```
from manimlib.imports import *
class AhojSvete(Scene):
    def construct(self):
        svetu = TextMobject("Ahoj, světe!")
        self.play(Write(svetu))
        self.wait()
```

Spouštíme: `python3 -m manim ukazky/ahoj-svete.py -p`

5.2. Lze zařadit TikZ?

Lze, ale síla linek se v svg nebere v potaz. Chce to experimentovat u konkrétního kódu. Zde je ukázka animace pro divadelní hru v souboru `ukazky/moda-live.py`, která nebyla použita, sloužila k testovacím účelům obrazovek a monitorů. Je vidět složitější animace, vsunutí textu do Ti*k*Zu, u proměnných užívám `r""`, abych nemusel zdvojovat zpětná lomítka.

```
from manimlib.imports import *
class ModaLive(Scene):
  def construct(self):
    text=r"\begin{tikzpicture}\node[draw=none]{Móda
     live!};\end{tikzpicture}"; modalive=TextMobject(text);
     modalive.scale(6); obdelnik=[""]*5
    obdelnik[0]=TextMobject(text); obdelnik[0].scale(2);
     obdelnik[0].to_edge(DOWN); obdelnik[0].set_color(YELLOW);
     obdelnik[0].rotate(PI)
    obdelnik[1]=TextMobject(text); obdelnik[1].scale(2);
     obdelnik[1].to_edge(UP); obdelnik[1].set_color(YELLOW)
    obdelnik[2]=TextMobject(text); obdelnik[2].scale(2);
     obdelnik[2].to_edge(LEFT); obdelnik[2].set_color(YELLOW);
     obdelnik[2].rotate(PI/2)
    obdelnik[3]=TextMobject(text); obdelnik[3].scale(2);
     obdelnik[3].to_edge(RIGHT); obdelnik[3].set_color(YELLOW);
     obdelnik[3].rotate(-PI/2)
    self.wait(1); self.play(Write(modalive),run_time=20); self.wait(1);
     self.play(FadeOut(modalive),run_time=5); self.wait(1)
    for x in [1,3,0,2]: self.play(Write(obdelnik[x]), run_time=2)
    modalive.set_color_by_gradient(RED,BLUE); self.wait(1);
     self.play(GrowFromCenter(modalive), run_time=5); self.wait(1);
     self.play(Transform(obdelnik[0], modalive),Transform(obdelnik[1],
     modalive), run_time=3); self.wait(1); self.play(Transform(obdelnik[2],
     modalive), Transform(obdelnik[3], modalive), run_time=3)
    for x in range(4): self.remove(obdelnik[x])
    self.play(FadeOut(modalive), run_time=5); self.wait(1);
     srdce=TextMobject(r"$\varheartsuit$"); srdce.scale(20);
```

```
    srdce.set_color(RED); srdce.rotate(-45);
    self.play(GrowFromCenter(srdce), run_time=5); self.play(FadeOut(srdce),
    run_time=2); self.wait(1); self.play(ShowCreation(modalive),
    run_time=40); self.wait(1); self.play(FadeOut(modalive), run_time=5);
    self.wait(1)
```

Animace: `python3 -m manim ukazky/moda-live.py -p`

5.3. Lze animovat matematiku?

Pozornější TEXisté si všimli, že v preambuli šablony je balíček halloweenmath, nyní jej použijeme v souboru `ukazky/rovnice.py`.

```
from manimlib.imports import *
class Rovnice(Scene):
  def construct(self):
    sum1=TextMobject(r"$$\sum_{i=0}^{n} {n\choose i} a^{n-i} b^i = (a+b)^n$$")
    sum1.scale(2); sum1.to_edge(UP); sum1.set_color(WHITE);
     self.play(Write(sum1), run_time=3); self.wait(1)
    sum2=TextMobject(r"$$\mathwitch*_{i=0}^{n} {n\choose i}
     \mathrightghost^{n-i} \mathleftghost^i \xrightwitchonbroom*{}
     (\mathrightghost+\mathleftghost)^n$$")
    sum2.scale(1.75); sum2.to_edge(DOWN); sum2.set_color(WHITE)
    self.play(ReplacementTransform(sum1.copy(), sum2), run_time=6);
     self.wait(4)
```

Animace: `python3 -m manim ukazky/rovnice.py -p`

5.4. Lze animovat kandži?

Asi nejrychlejší způsob je si v `manimlib/constants.py` zapnout ChineseTEX:

```
TEX_USE_CTEX = True # False
```

Tím si zajistíme, že budeme užívat `xelatex` a TEXovou šablonu v souboru `manimlib/ctex_template.tex`. Tu jsem si po odzálohování upravil do této podoby:

```
\documentclass[preview]{standalone}
\usepackage{amsmath,amssymb}
\usepackage[UTF8]{ctex}
\begin{document}
YourTextHere
\end{document}
```

Jednoduchá ukázka by mohla vypadat takto:

```
from manimlib.imports import *
class Japonstina(Scene):
  def construct(self):
    sum1=TextMobject(r"今日は何曜日ですか。")
    sum1.scale(2.5); sum1.to_edge(UP); sum1.set_color(RED);
     self.play(Write(sum1), run_time=3); self.wait(1)
    sum2=TextMobject(r"３日です。")
    sum2.scale(6); sum2.to_edge(DOWN); sum2.set_color(RED)
    self.play(ReplacementTransform(sum1.copy(),sum2),run_time=5); self.wait(2)
```

Animaci získáme přes: `python3 -m manim ukazky/japonstina.py -p`

5.5. Lze animovat výstupy z teorie grafů?

Na pomoc jsem si vzal knihovnu manimnx a vypnul jsem si užití CTEXu:

```
TEX_USE_CTEX = False
```

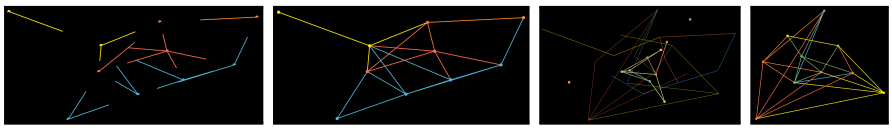
Doinstaloval jsem si potřebné:

```
$ git clone https://github.com/rajatvd/manimnx
$ sudo -H pip3 install networkx==2.3
```

V `manimnx/example.py` jsem zasáhl do jednoho řádku tímto způsobem, protože `manim.py` načítám z pracovního adresáře:

```
import manimnx.manimnx.manimnx as mnx
```

Ukázka: `python3 -m manim manimnx/example.py RandomGraphs -p`



5.6. Lze animovat diagramy?

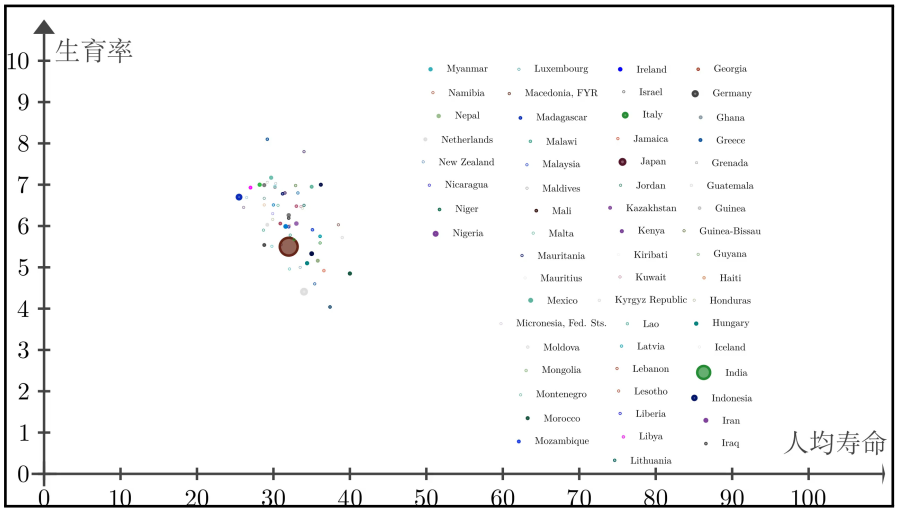
Jako poslední ukázku jsem vybral extrémní případ knihovny Danim.

```
$ git clone https://github.com/graviton1221/Danim
$ sudo -H pip3 install pandas
```

V `Danim/BubbleChart/bubblechart_constant.py` jsem \\ upravil na /, jak pracuji pod Linuxem. A ještě jednou jsem si zapnul CTEX.

Každý popisek se generuje do zvláštního TEXového souboru, vznik animace tedy trvá: `python3 -m manim Danim/BubbleChart/BubbleChartAnimation.py BubbleChartAnimation -p`

Ukázka z 19. vteřiny videa v inverzních barvách kvůli tisku bulletinku.



6. Co s programem Manim neumím?

Co jsem nepotřeboval a nezkoušel jsem do hloubky je živý přenos přes protokol tcp (parametr `--livestream`) s podporou pro Twitch přes parametry `--to-twitch` a `--with-key`. Hashovací klíč jsem našel na twitch.tv pod uživatelem; Settings; Channel and Videos a pak Primary Stream key (Copy nebo Show).

Práce se soubory svg je nativní, zde jsou oblíbené figurky přednášejících (PíCreature, Stickman a Linus). Lze si vytvořit vlastního průvodce. Nezkoušel jsem, byť návod existuje, https://talkingphysics.wordpress.com/2018/08/14/working-with-svg-files-manim-series-part-12/.

Vzniká podpora pro JavaScript, viz github.com/JazonJiao/Manim.js. Přes JavaScript bych na animace volil spíš d3js.org.

Vzniká též webové rozhraní, viz github.com/eulertour/eulerv2. To mi přišlo ve webovém prohlížeči extrémně pomalé.

Pokud chcete nahlédnout na animace zmíněné v článku, a programátorské a TEXové věci si nezkoušet, navštivte můj skromný kanál na YouTube:

youtube.com/playlist?list=PLnD-4Ssyh8yOQi08n9L8WJ3wV3u2p9VO9

Co zmínit závěrem? Je to jiný svět, snad jsem vám hlavu nezamotal víc než je zdrávo, mám totiž před sebou víc otázek než odpovědí, i tak však přeji hezké bádání s nástrojem Manim!

- Můžeme užít LuaLaTeX a libovolné písmo přes balíček fontspec?
- Lze získat animaci jako sérii souborů svg?
- Může se objekt pohybovat po libovolné křivce, např. po spirále?
- Je možné (de)aktivovat vyhlazování písem?
- Jak lze ideálně zařadit grafiku z programů METAPOST a PSTricks?
- Jak by to bylo se zařazením 3D objektů (Asymptote, Blender)?
- Lze vložit grafické výstupy z programů typu Matplotlib a R?
- Lze vložit do vznikající animace video? Například ve videu o kvaternionech je pohyb ruky, tedy to lze, ale jak ...

Je vidět, že 3blue1brown je učitel tělem i duší a nutí nás přemýšlet a programátorsky hledat a experimentovat. Ne nadarmo byl 13. 3. 2020 Grant Sanderson pozván jako řečník do TEDxBerkeley (https://youtube.com/watch?v=s_L-fp8gDzY) s tématem *What Makes People Engage With Math*, minimálně jako duševní kantorská podpora v době koronaviru.

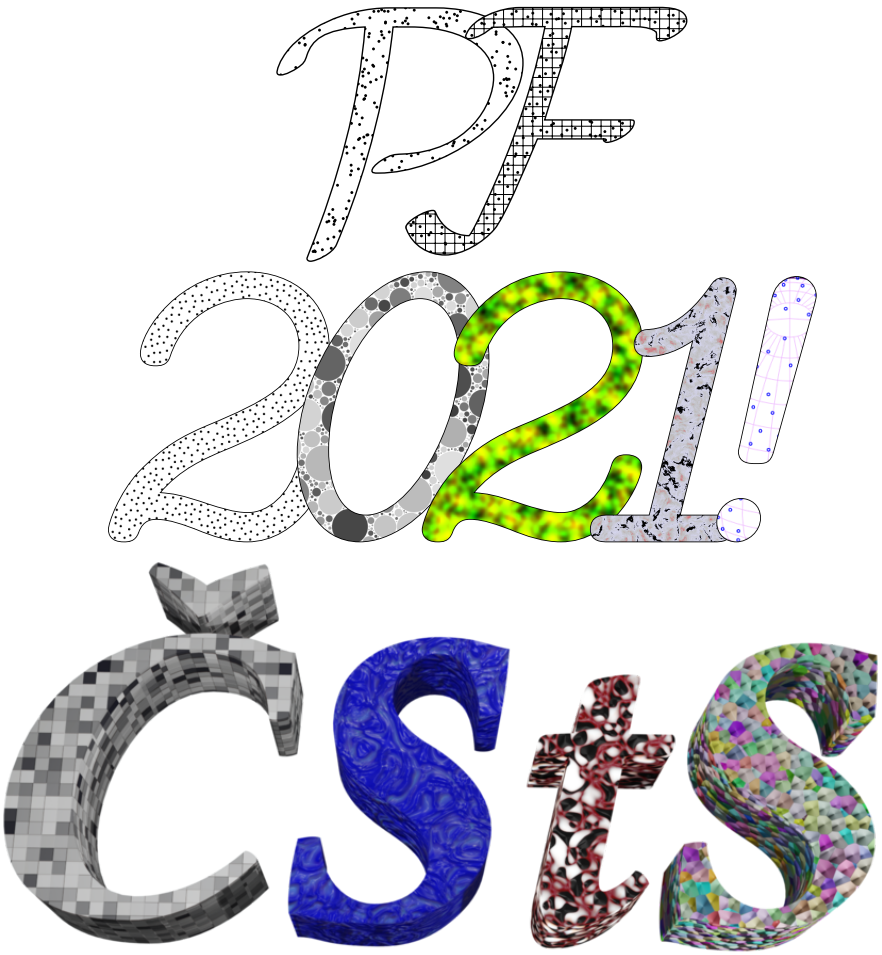
> *... if you have a soul, you have to know why, right?*
> — Grant Sanderson @ 00:13:04,159265

PF2021! ANEB NENÍ ŠUM JAKO ŠUM
PF2021! OR DIFFERENT TYPES OF NOISE

Pavel Stříž

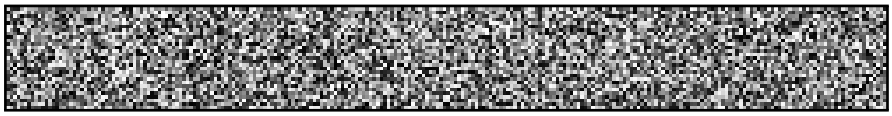
E-mail: pavel@striz.cz

Motto: [...] 12585 33893 43498 [...], str. 165, řádek 08210, sloupce 5–7.
A Million Random Digits with 100,000 Normal Deviates



1. Úvodem pár vzpomínek

Mé první vzpomínky na práci s náhodností spadají pod Sinclair BASIC na základní škole, kdy jsme např. volili den v týdnu jako jedno z čísel 1 až 7 a dál s tím pracovali. Pár let poté jsem podobné příklady procvičoval v QBASICu s otevřenou knihou *Sbírka úloh z programování* od manželů Töpferových. V průběhu času se člověk dostal k fyzice a šumu u televizí až ke kosmickému záření.



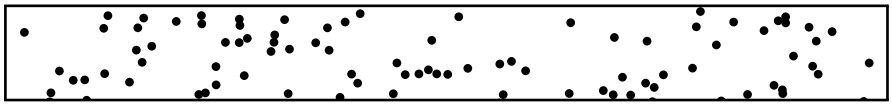
Zaujala mě práce Norberta Wienera, nejen jeho aplikace zpětné vazby, ale i práce nad Brownovým pohybem. To jsou ty známé základoškolské fyzikální pokusy vhození hypermanganu do vody. Dnes je bílý šum (angl. white noise) brán vážně ve všech významnějších oblastech přírodovědeckého bádání, včetně ekonomie, informatiky, statistiky a geometrie.

Z poslední doby se mi do ruky dostala kniha Davida Johnstona *Random Number Generators—Principles and practices* z roku 2018, kde se to hemží kódy v C a Pythonu. Dle MSC2020 bychom naši článkovou rešerši začali asi v oblastech 60H40 (White noise theory), 65Cxx (Probabilistic methods...) či 11K45 (Pseudo-random numbers; Monte Carlo methods).

2. PF aneb Základ černobíle

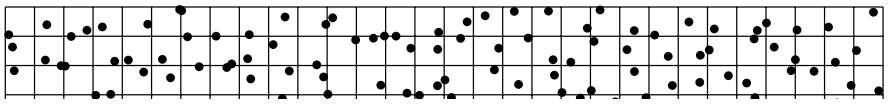
Na první ukázce, písmenku $\mathcal{P}$, vidíme typickou situaci užití pseudonáhodných čísel ve 2D (angl. pseudo-random numbers). Bez ohledu na kvalitu statistických vlastností to není po vizuální stránce příjemné. Jsou tam prázdná oka, kruhy se mohou překrývat. K testování lze doporučit `www.random.org`. Zde je ukázka přes Ti*k*Z.

```
\documentclass{standalone}
\usepackage{tikz}
\tikzset{inner sep=0pt, outer sep=0pt}
\begin{document}
\begin{tikzpicture}
\foreach \x in {1,...,900} {
  \pgfmathparse{rnd*100} \let\malx=\pgfmathresult
  \pgfmathparse{rnd*100} \let\maly=\pgfmathresult
  \node[xshift=\malx mm, yshift=\maly mm, circle, fill, minimum width=1mm]{};
} % end of \foreach \x
\end{tikzpicture}
\end{document}
```



Druhou stranou mince by byla dokonalá mřížka z bodů. Spojení obou nápadů vzniká stratifikovaný výběr (angl. supersampling či jittered grid). Prvně si plochu rozdělíme na menší čtverce a v každém volíme po jednom bodu. Chceme-li bodů víc, zjemníme mřížku. Dostáváme písmenko $\mathcal{F}$. Vizuálně je to lepší, ale stále tam jsou místy body u sebe a občas řeky. To vadí především typografům z čteného textu odstavců. U mřížky se mohou objekty překrývat. Opět vzorek přes Ti*k*Z.

```
\documentclass{standalone}
\usepackage{tikz} \tikzset{inner sep=0pt, outer sep=0pt}
\begin{document}
\begin{tikzpicture}
\draw[step=3.3333mm] (0,0) grid (100mm,100mm);
\foreach \x in {1,...,30} {
  \foreach \y in {1,...,30} {
    \pgfmathparse{3.3333*(rnd-1+\x)} \let\malx=\pgfmathresult
    \pgfmathparse{3.3333*(rnd-1+\y)} \let\maly=\pgfmathresult
    \node[xshift=\malx mm, yshift=\maly mm, circle, fill, minimum
     width=1mm]{};
  } % end of \foreach \y
} % end of \foreach \x
\end{tikzpicture}
\end{document}
```

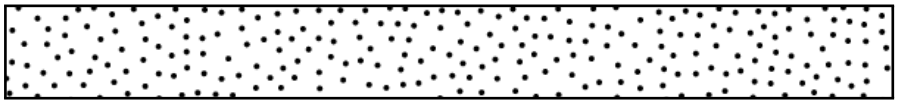


3. 2021 aneb Přes stupně šedi do barvy

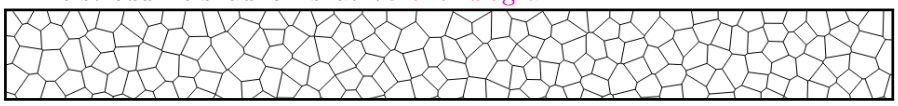
Grafici šli dál.

3.1. Robert Bridson

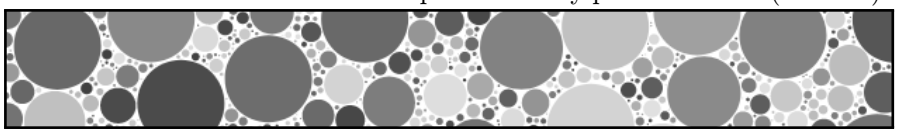
Jason Davies (zde, je znám hlavně díky mrakům slov) či Mike Bostock (zde, zakladatel serverů `bl.ocks.org` a `observablehq.com`) představují Bridsonův algoritmus (2007) generování bodů s geometrickým vztahem, že žádný nový bod nesmí být blíž než určená vzdálenost (angl. Poisson-disc sampling). Nelze již mluvit o pokusu náhodného generování, neb existuje mezi body vztah. Vizuálně je to pro lidské oko příjemné. Ukázka je v první číslici *2*. Davies vykresluje body přes canvas HTML5, Bostock do svg přes `D3js`.



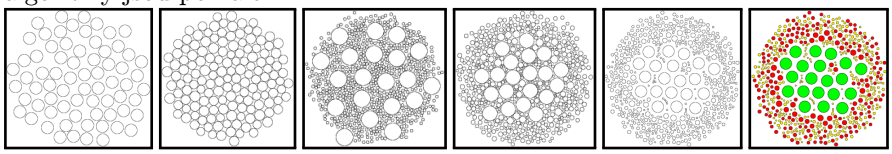
Ze středů lze snadno získat Voronoi diagram.



Zobecnění přináší algoritmus Mitchell's best-candidate, kdy se generuje sada k bodů a vybírá se z nich jen jeden, takový, který je nevzdálenější vůči všem ostatním. To dává možnost např. volit různý poloměr kruhů (číslice *0*).



Zájemci mohou nahlédnout na mé starší pokusy přes Lua, byť tedy užité algoritmy jsou pomalé.

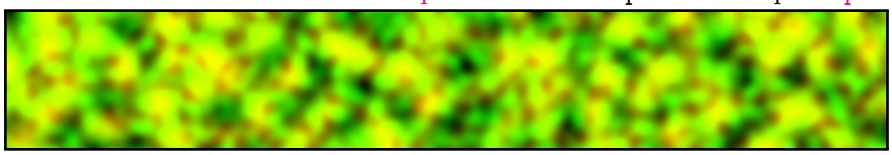


3.2. Ken Perlin

Mezník představuje Perlinův šum (angl. Perlin noise) z roku 1983 (příspěvek Kena Perlina z roku 1985 se jmenuje *An Image Synthesizer*), kdy dochází k interpolaci mezi body. Tím lze snadno vytvářet šum ve 2D a 3D, se zahrnutím času či barvy i ve vyšších rozměrech. Zde je typická ukázka, vypadá to jak výškový model (angl. DEM) známý z geografických inf. systémů.

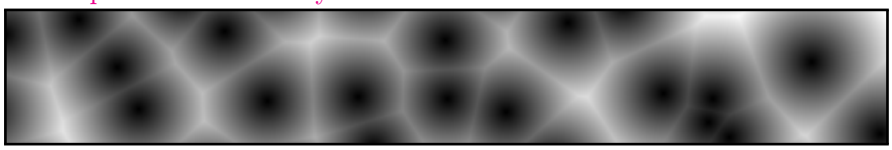
Za pozornost stojí i článek Ken Perlin a Fabrice Neyret: *Flow Noise.* Podobné výsledky dostáváme pomocí algoritmu Simplex noise a volně dostupné varianty OpenSimplex noise. Zde jsou dostupné implementace v Javě, JavaScriptu a nezapomínejme na Rust. Zaujal mě článek *Recursive Wang Tiles for Real-Time Blue Noise.* Pro studenty lze doporučit na YouTube kanál The Coding Train Daniela Shiffmana v jeho oblíbeném nástroji `p5js` a jeho knihu *The Nature of Code.*

Druhá cifra *2* vznikla v JavaScriptu v balíčku `simplex-noise` přes `npm`.

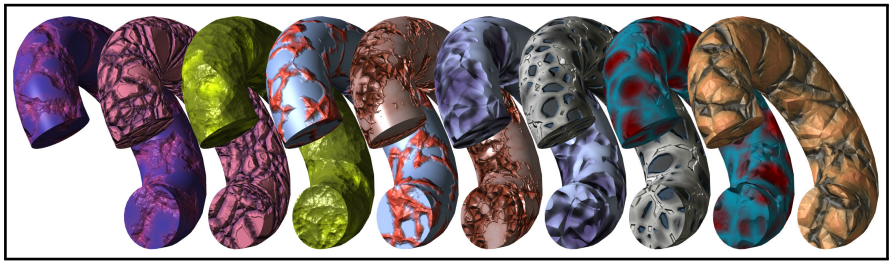


3.3. Steven Worley

Další skok přichází v roce 1996, kdy Steven Worley na konferenci představuje tvorbu procedurální textury.



Zde jsou ukázky z jeho článku *A cellular texture basis function.*

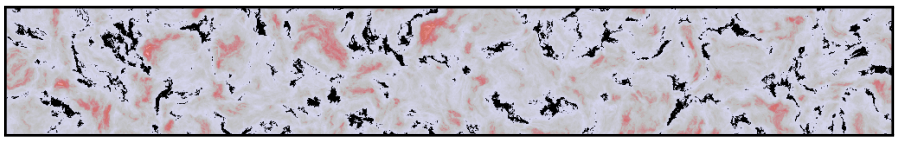


Poslední cifra, *1*, je z `examples/texturegranite.rs` z knihovny `noise` v0.6.0 na `crates.io`, konkrétně soubor `texture_granite_planar.png`. Získáváme malbu skoro jako od Jacksona Pollocka.

Po instalaci:

```
$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
$ echo "export PATH=$HOME/.cargo/bin:$PATH" >>~/.bashrc
$ source $HOME/.cargo/env
$ git clone https://github.com/Razaekel/noise-rs.git
$ cd noise-rs
$ cargo build
```

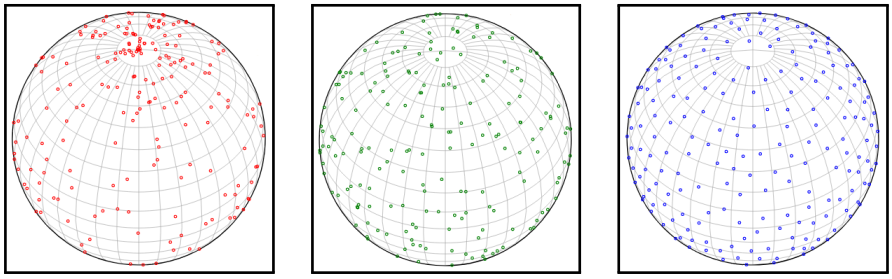
jsem spouštěl `cargo run --example texturegranite`.



Vážnějším zájemcům doporučuji knihu Patricio Gonzales Vivo a Jen Lowe: *The Book of Shaders* se zvýrazněnými ukázkami. Kniha vzniká od roku 2015 a je stále v přípravě a ve vývoji v OpenGL ES. Vážným zájemcům pak doporučuji knihy Ebert, Musgrave, Peachey, Perlin a Worley: *Texturing and Modeling: A Procedural Approach,* 3. vyd. z roku 2002 a z roku 2017 Tanya X. Short a Tarn Adams: *Procedural generation in game design.*

4. Koule na místo vykřičníku

Rust má v ukázkách příklad vzniku textury pasovanou na kouli. Je to předchozí kód, další vygenerovaný soubor `texture_granite_sphere.png` ve složce `example_images`. Rovnovnoměrné rozdělení bodů na kouli je známý a vyřešený problém, viz MathWorld. Jason Davies srovnává intuitivní řešení ve sférické soustavě souřadnic (vlevo), řešení s korekcí (uprostřed) a aplikovaný Mitchellův algoritmus výběru nejlepšího kandidáta (koule vpravo). Charakteristikami se dostáváme na hranici modrého šumu.



5. ČStS aneb Vzorky pomocí procedurálních textur

Situace se komplikuje, pokud zvolíme obecný 3D objekt.

Existuje řada programů, které umí pracovat s texturami. Mezi nejvýraznější svobodné programy patří Blender (zkr. BS). Ten je mezi námi už od roku 1988, jen o 10 let mladší než TEX (1978) a o 5 let starší než R (1993). Některé postupy jsou vlastní, některé inspirované jinými programy na 3D grafiku. Jednou z inspirací byl program Filter Forge, kde se užívá jazyk Lua.

Za pozornost dávám knihu Richarda Egdahla *Texture Magic: Procedural Textures for Blender Cycles,* kterou považuji za představitele Blenderu do verze 2.79b. Důležité je, že Blender je specialista na 3D a položení textur na libovolné 3D objekty je přirozený krok grafiků.

Blender má nadstavbu Sverchok (zkr. SV, rusky сверчок) na parametricky definované 3D objekty. Příchod nadstavby Animation Nodes (zkr. AN) znamená mezník u animování s následnou možností úpravy textů.

Blender udělal obří skok od verze 2.80 s tahem k nástroji Everything Nodes, kdy přes grafické rozhraní (angl. shader nodes) by měly být dostupné téměř všechny nástroje Blenderu, speciálně systém částic.

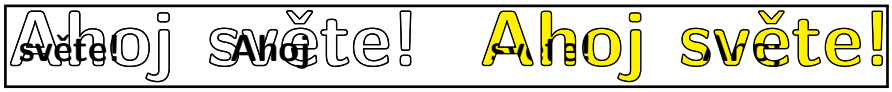
Lze doporučit YouTube kanál LiveNoding od Jimma Gunawaneho.

6. Kompletace novoročenky

Znaky v ČStS jsou vytvořené v Blenderu za pomocí procedurálních textur a vyrenderovány jako rastrové obrázky: *Č* přes White Noise Texture, *S* přes Noise Texture, *t* přes Musgrave Texture a *S* pomocí Voronoi Texture.

Pro TEXisty bude zajímavá první část. Přes Ti*k*Z vkládám pozadí dovnitř znaků. Zde je minimální ukázka „Ahoj světe!“ bez a s vyříznutím. Dávám na sebe pozadí velkých slov, malá slova a obrys velkých znaků.

```
\documentclass{article}
\pagestyle{empty}
\usepackage{tikz}
\begin{document}
\def\ukaz#1#2#3#4{%
\begin{tikzpicture}[text height=2ex, text depth=1ex]
\node{\Huge\bfseries\sffamily
  \pgfsetstrokecolor{black}\pgfsetfillcolor{yellow}%
  \pdfextension literal {#1 Tr}%
  \makebox[0pt][l]{#2}%
  \makebox[0pt][l]{\large
    \pdfextension literal {#3 Tr}%
    \tikz[trim left, baseline=0mm]
      \tikz{\node[xshift=7mm, yshift=1.5mm, text height=2ex, text
     depth=1ex]{\color{black}#4};
      };% end of \tikz
  }% end of \makebox
  \pdfextension literal {1 Tr 0.4 w}%
  \makebox[0pt][l]{#2}%
  };% end of \node
\end{tikzpicture}%
}% end of \ukaz
\ukaz{1}{Ahoj}{0}{světe!}\kern2cm \ukaz{1}{světe!}{0}{Ahoj}\kern28mm
\ukaz{6}{Ahoj}{0}{světe!}\kern2cm \ukaz{6}{světe!}{0}{Ahoj}
\end{document}
```



[...] but with Perlin noise I may pick numbers like this: 2, 3, 4, 3, 4, 5, 6, 5, 4, 5, 6, 7 [...] Daniel Shiffman @ Perlin Noise and Flow Fields

test1

test2