The ML/DL/HPC Ecosystem of the HybriLIT Heterogeneous Platform (MLIT JINR) for Applied Research

Zuev M.I. [1], Butenko Yu.A. [1], Nechaevskiy A.V. [1], Podgainy D.V.[1], Rahmonov I.R.[2], Streltsova O.I.[1]

[1] Meshcheryakov Laboratory of Information Technologies

[2] Bogoliubov Laboratory of Theoretical Physics

This work was supported by the Russian Science Foundation under grant No 22-71-10022

The XXVI International Scientific Conference of Young Scientists and Specialists (AYSS-2022)

Dubna, JINR, 24 – 28 October 2022

Ecosystem for ML/DL/HPC tasks

HPClab component

High **P**erformance **C**omputing

VM with JupyterHub and SLURM [**`https://jlabhpc.jinr.ru`**]

- ❑ Intel Xeon Gold 6126 (24 Cores @ 2.6 GHz)
- ❑ 32 GB RAM

Educational component

JupyterLab Server [**`https://studhub.jinr.ru`**]

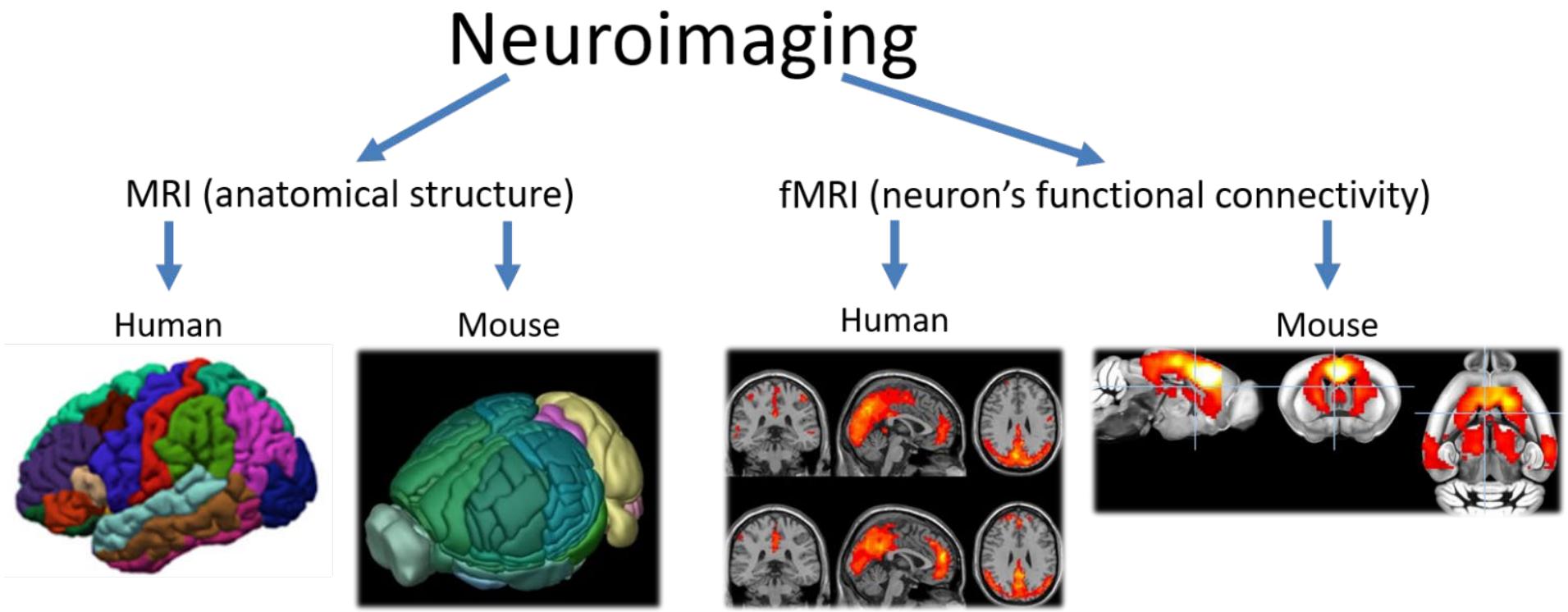
- ❑ 2x Intel Xeon Gold 6152 (22 Cores @ 2.1 GHz)
- ❑ 512 GB RAM

Computation component

Server with NVIDIA Volta [**`https://jhub2.jinr.ru`**]

- ❑ 2x Intel Xeon Gold 6148 (20 Cores @ 2.4 GHz)
- ❑ 4x NVIDIA Tesla V100 SXM2 32 GB HBM2
- ❑ 512 GB RAM

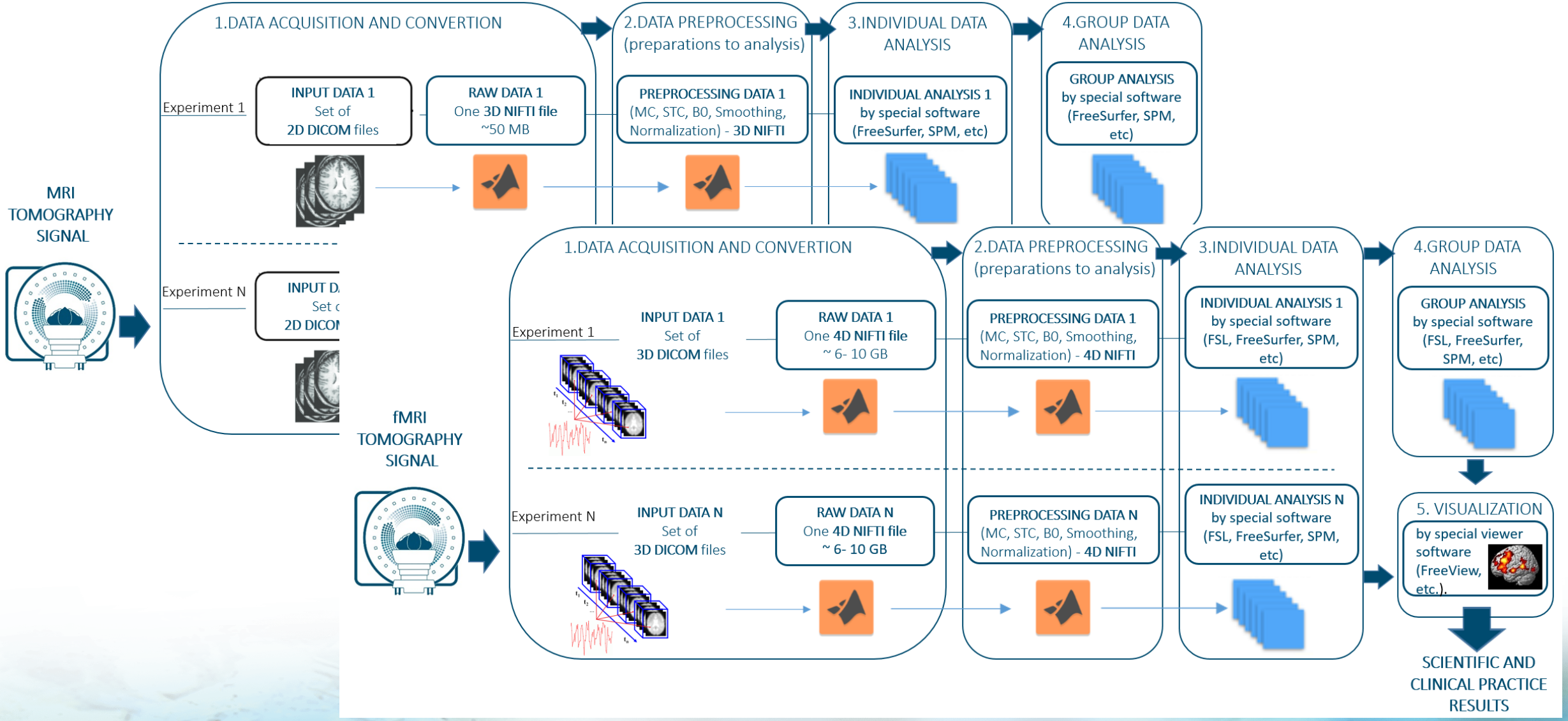
Workflow for Analyzing Experimental MRI/fMRI Data
Scientific field



The study of the anatomical structure and functional neuronal activity of the brain is an actual task of modern neurobiology. At present, MRI and fMRI are considered the most effective methods of neuroimaging. The modern equipment allows conducting non-invasive experiments of these types for both humans and laboratory animals (monkeys, mice, etc.). The discovery of these methods has provided new opportunities for neurobiologists. In particular, the use of MRI/fMRI methods enables long-term research of the brain, human or laboratory animals.

Workflow for Analyzing Experimental MRI/fMRI Data

Experiments data work flows



Workflow for Analyzing Experimental MRI/fMRI Data
Software and work installed packages

Algorithm for users

1. Authorization in the system
2. Downloading the DICOM catalog
3. Converting the DICOM directory to a NIFTI file and a TXT experiment metadata file
4. Processing the NIFTI file by specialized software (FreeSurfer or FSL)
5. Visualization of results

List of installed packages

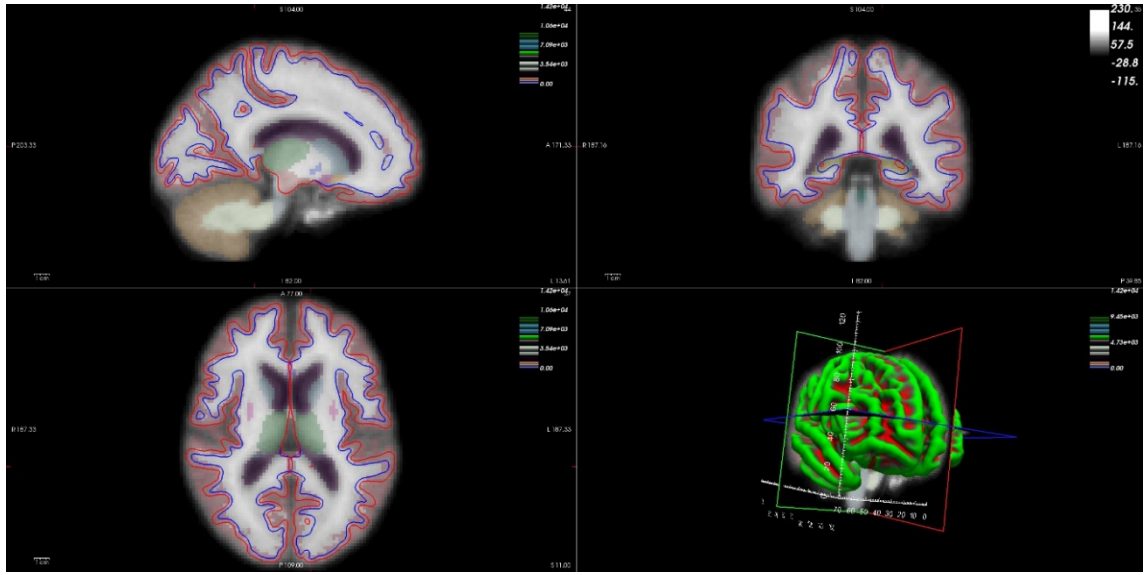
- ❑ **MRI Convert** for converting the NIFTI file from the original set of DICOM files;
- ❑ **FreeSurfer** for NIFTI file processing;
- ❑ **FreeView** for building images after NIFTI file processing;
- ❑ **FSL**, alternative package for NIFTI file processing;
- ❑ **FSLeyes** (viewer from the FSL package) for building images after NIFTI file processing.

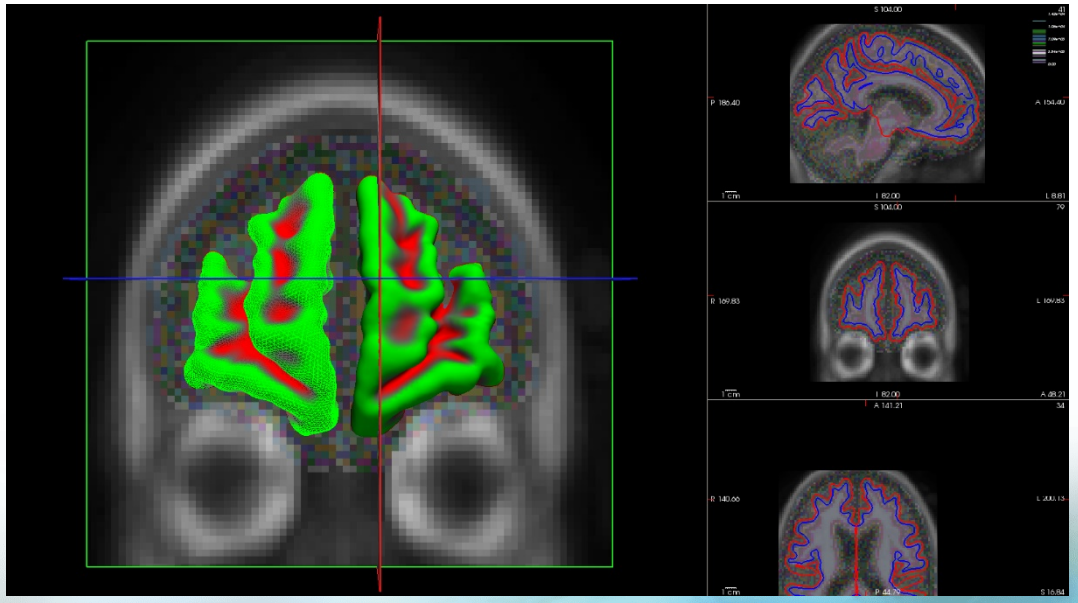
Using the parallelization, the processing time was reduced by 3 times.

Workflow for Analyzing Experimental MRI/fMRI Data

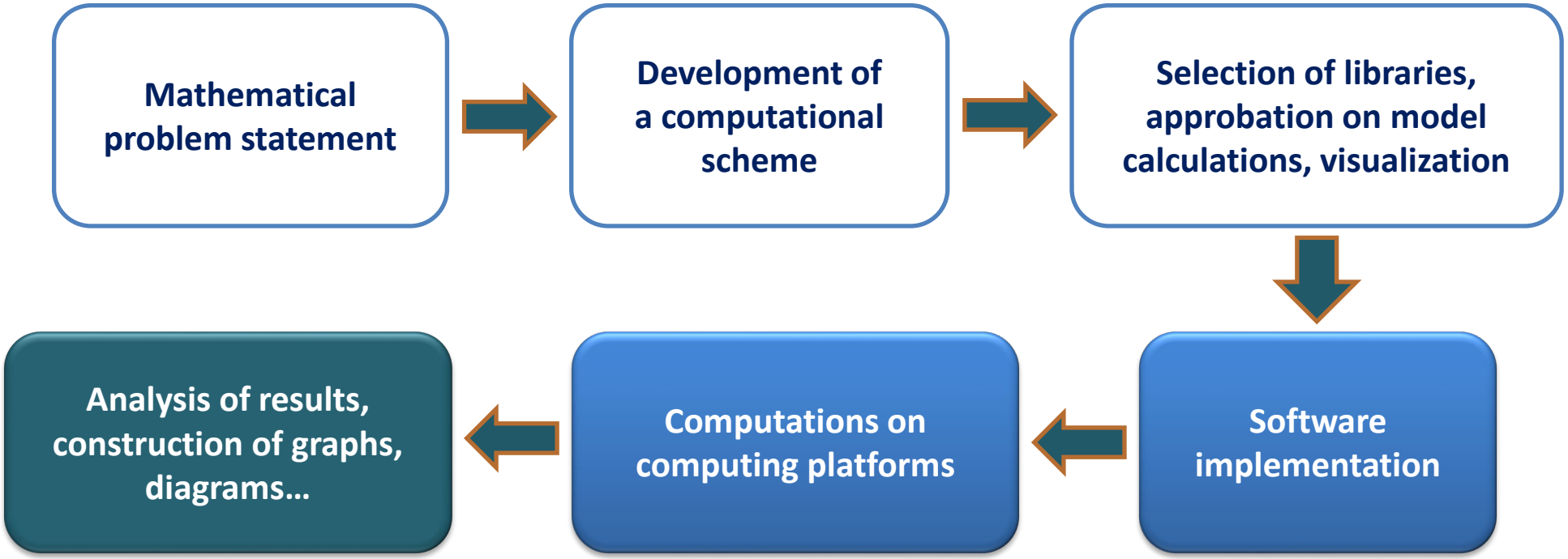
Data processing results

Index	SegId	NVoxels	Volume_m	StructName	normMean	normStdD	normMin	normMax	normRang
1	4	7424	7766.0	Left-Lateral-Ventricle	26.6416	12.7145	8.0000	88.0000	80.0000
2	5	219	279.7	Left-Inf-Lat-Vent	48.2466	13.1143	16.0000	77.0000	61.0000
3	7	12760	13586.0	Left-Cerebellum-White-Matter	88.9705	6.4768	31.0000	106.0000	75.0000
4	8	53201	52897.8	Left-Cerebellum-Cortex	63.6782	11.4213	14.0000	99.0000	85.0000
5	10	7087	6882.0	Left-Thalamus-Proper	92.4115	10.6040	36.0000	114.0000	78.0000
6	11	2909	2902.5	Left-Caudate	77.1495	8.2775	34.0000	100.0000	66.0000
7	12	3929	3877.7	Left-Putamen	88.1733	5.8184	61.0000	109.0000	48.0000
8	13	1818	1807.4	Left-Pallidum	101.6139	4.6422	62.0000	117.0000	55.0000
9	14	1081	1135.9	3rd-Ventricle	30.9713	14.0876	2.0000	79.0000	77.0000
10	15	2292	2372.1	4th-Ventricle	25.8233	11.4535	1.0000	81.0000	80.0000
11	16	20077	20065.1	Brain-Stem	81.7819	10.6608	16.0000	117.0000	101.0000
12	17	3744	3671.1	Left-Hippocampus	70.0577	8.0581	31.0000	104.0000	73.0000
13	18	1143	1115.6	Left-Amygdala	73.6763	6.9137	51.0000	93.0000	42.0000
14	24	1176	1163.5	CSF	40.7985	14.0558	11.0000	82.0000	71.0000
15	26	304	302.1	Left-Accumbens-area	76.9243	7.8175	44.0000	96.0000	52.0000
16	28	3805	3648.3	Left-VentralDC	94.5396	11.7385	26.0000	119.0000	93.0000
17	30	8	17.4	Left-vessel	68.8750	2.7484	65.0000	73.0000	8.0000
18	31	534	477.5	Left-choroid-plexus	52.5730	14.0221	24.0000	84.0000	60.0000
19	43	8748	9056.6	Right-Lateral-Ventricle	26.3001	12.3941	5.0000	93.0000	88.0000
20	44	267	291.8	Right-Inf-Lat-Vent	36.9288	13.8134	8.0000	69.0000	61.0000
21	46	12509	13351.8	Right-Cerebellum-White-Matter	84.6867	6.0372	31.0000	103.0000	72.0000
22	47	54192	54030.3	Right-Cerebellum-Cortex	62.9141	10.6152	13.0000	101.0000	88.0000
23	49	6480	6275.9	Right-Thalamus-Proper	89.9441	9.7863	26.0000	114.0000	88.0000
24	50	3249	3159.0	Right-Caudate	77.1265	8.7631	42.0000	100.0000	58.0000
25	51	4072	4061.9	Right-Putamen	85.7665	6.4403	61.0000	106.0000	45.0000
26	52	1764	1721.9	Right-Pallidum	99.7528	4.3327	68.0000	113.0000	45.0000
27	53	3738	3736.3	Right-Hippocampus	69.7105	7.4139	37.0000	102.0000	65.0000
28	54	1224	1180.6	Right-Amygdala	69.0556	6.6842	50.0000	91.0000	41.0000
29	58	373	355.8	Right-Accumbens-area	75.5952	6.0272	47.0000	93.0000	46.0000
30	60	3863	3649.7	Right-VentralDC	92.7108	11.3672	36.0000	118.0000	82.0000
31	62	15	32.0	Right-vessel	63.2000	4.4272	55.0000	71.0000	16.0000
32	63	718	622.3	Right-choroid-plexus	51.1267	12.1938	22.0000	78.0000	56.0000
33	72	0	0.0	5th-Ventricle	0.0000	0.0000	0.0000	0.0000	0.0000
34	77	1096	940.1	WM-hypointensities	69.0557	12.3296	34.0000	109.0000	75.0000
35	78	0	0.0	Left-WM-hypointensities	0.0000	0.0000	0.0000	0.0000	0.0000
36	79	0	0.0	Right-WM-hypointensities	0.0000	0.0000	0.0000	0.0000	0.0000
37	80	0	0.0	non-WM-hypointensities	0.0000	0.0000	0.0000	0.0000	0.0000
38	81	0	0.0	Left-non-WM-hypointensities	0.0000	0.0000	0.0000	0.0000	0.0000
39	82	0	0.0	Right-non-WM-hypointensities	0.0000	0.0000	0.0000	0.0000	0.0000
40	85	127	126.0	Optic-Chiasm	83.3780	19.0929	27.0000	123.0000	96.0000
41	251	1150	1044.2	CC_Posterior	99.6174	17.6343	38.0000	124.0000	86.0000
42	252	626	540.0	CC_Mid_Posterior	88.8578	18.7974	14.0000	116.0000	102.0000
43	253	494	433.3	CC_Central	93.6943	17.0553	37.0000	113.0000	76.0000
44	254	725	658.9	CC_Mid_Anterior	94.8083	16.3893	37.0000	127.0000	90.0000
45	255	907	853.0	CC_Anterior	102.6648	12.9003	28.0000	124.0000	96.0000





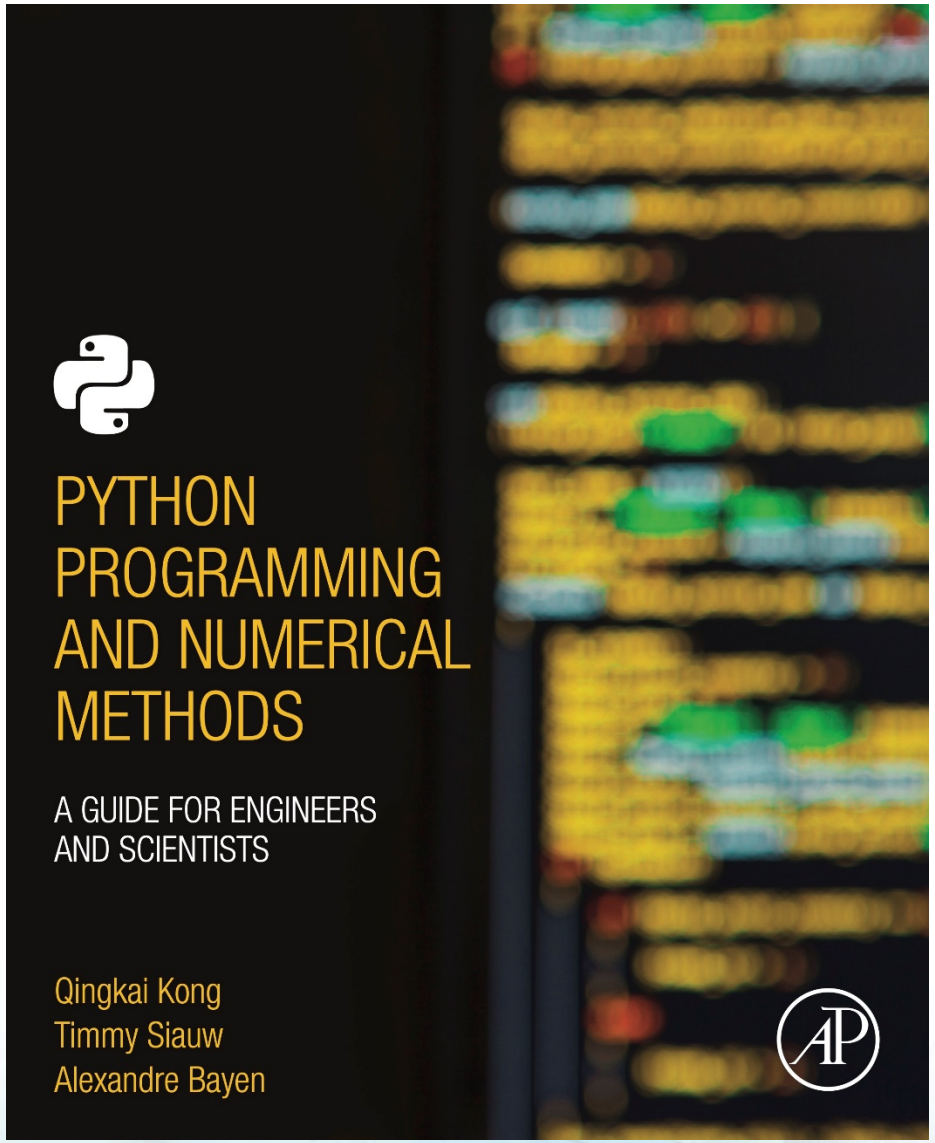
Numerical research process



The creation of a toolkit that allows one to carry out computations, to visualize the results within a single application, and perform the most resource-intensive calculations in parallel is an urgent task.
The *Jupyter Notebook* environment provides this capability.

Python Numerical Methods

pythonnumericalmethods.berkeley.edu



jupyter {book}

jupyter

Keras

matplotlib

pandas

NumPy

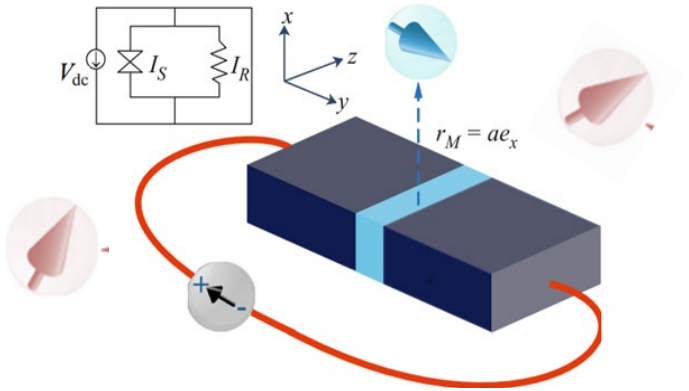
seaborn

Joblib

Development of a package for system research «JJ-nanomagnets»

jupyterhub

Symbolic computations block



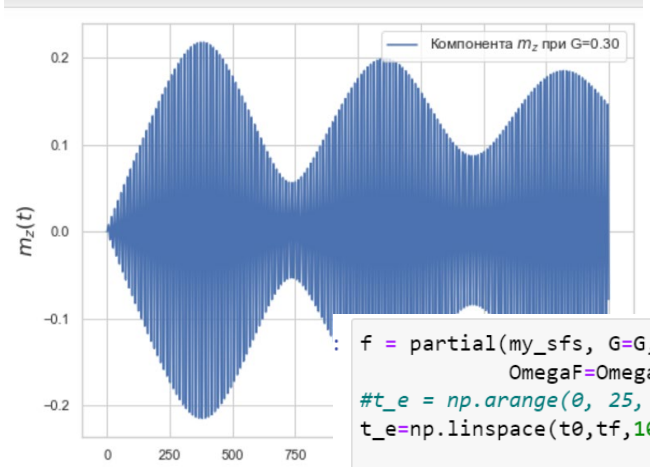
$$\gamma_{m_i} \quad = \quad = -\frac{\mu_0}{2\Phi_0}\int d\mathbf{r}_i \frac{\mathbf{M}_i \times \mathbf{r}_i}{r^3}$$

$$B_{12}(r_{12}, m_1) = \frac{\mu_0}{4\pi}\left(\frac{3(m_1 \cdot \hat{r})\hat{r}}{b^5} - \frac{m_1}{b^3}\right)$$

SymPy is a Python library for symbolic mathematics.

matplotlib is a main library for building graphs, diagrams in Python.

Block of numerical computations and analysis




```
f = partial(my_sfs, G=G, alpha=alpha, k=k,  \
            OmegaF=OmegaF, V=V)
#t_e = np.arange(0, 25, 0.0001)
t_e=np.linspace(t0,tf,100000)

s0 = np.array([0, 1, 0])
sol_1=solve_ivp(f,[t0,tf],s0, t_eval=t_e, method='RK45')
```

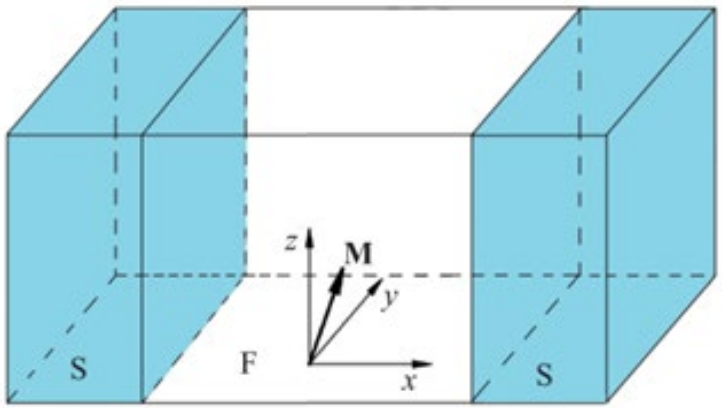
SciPy is an open-source software for mathematics, science, and engineering.

Parallelization of multi-parameter computations

Joblib is a set of tools to provide lightweight pipelining in Python

Problem to study the dynamics of magnetization in a Phi-0 Josephson Junction (SFS structure)

Collaboration with Ilhom Rahmonov (Bogoliubov Laboratory of Theoretical Physics, JINR)



The dynamics of the magnetic moment M of the system under consideration is described by the Landau-Lifshitz-Gilbert equation:

$$\frac{dm_x}{dt} = -\frac{1}{1+M^2\alpha^2}\{m_y H_z - m_z H_y + \alpha[m_x(M,H) - H_x]\},$$
$$\frac{dm_y}{dt} = -\frac{1}{1+M^2\alpha^2}\{m_z H_x - m_x H_z + \alpha[m_y(M,H) - H_y]\}$$
$$\frac{dm_z}{dt} = -\frac{1}{1+M^2\alpha^2}\{m_x H_y - m_y H_x + \alpha[m_z(M,H) - H_z]\},$$

$M = [m_x, m_y, m_z]$ are the magnetic moment components; the effective field components $H = [H_x, H_y, H_z]$ depend on the Josephson phase difference ϕ and are defined as follows:

$$H_x(t) = 0,$$
$$H_y = Gr\sin(\phi(t) - tm_y(t)),$$
$$H_z(t) = m_z(t).$$

The equation for the Josephson phase difference $\phi(t)$ is determined from the equation for the electric current I flowing through the Josephson junction, measured in units of the critical current I_c:

$$\frac{d\phi}{dt} = -\frac{1}{w}\left(\sin(\phi - rm_y) + r\frac{dm_y}{dt}\right) + \frac{1}{w}I,$$

Model parameters:
G – ratio of the Josephson energy to the magnetic anisotropy energy;
r – spin-orbit interaction constant;
α – Hilbert dissipation parameter;
in this study $w = 1$.

Python implementation

Calculations for different values of parameters

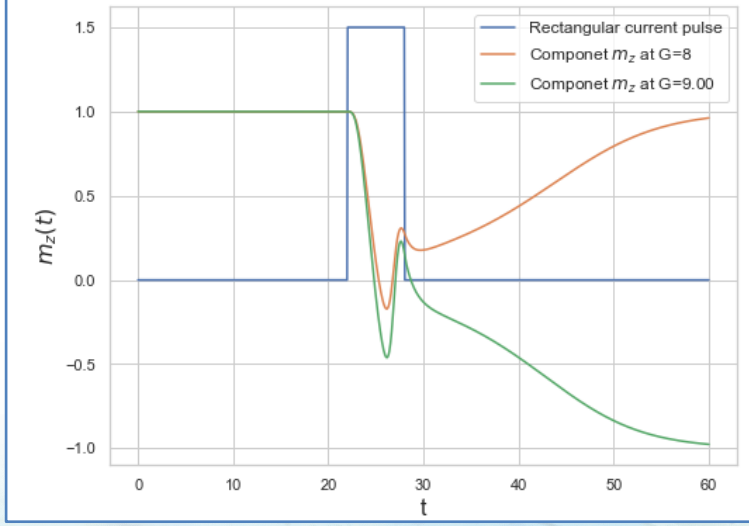
To analyze the possibility of reversing the magnetic moment of the ϕ_0-Josephson junction at different values of the parameters, we will carry out calculations for G=8.9.

```
from scipy.integrate import solve_ivp
from functools import partial
```

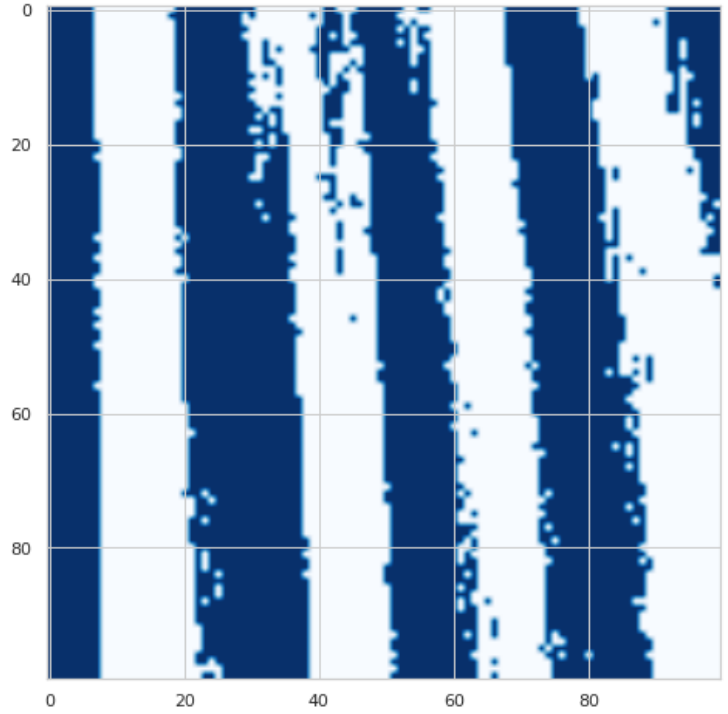
```
G=9
f = partial(my_sfs,  G=G, r=r, alpha=alpha, \
                As=As, t_s=t_s, delta_t=delta_t)
#t_e = np.arange(0, 25, 0.0001)
t_e=np.linspace(0,60,1000)

s0 = np.array([0, 0, 1, 0])
sol_2=solve_ivp(f,[0,60],s0, t_eval=t_e) # method = 'Radau'
```

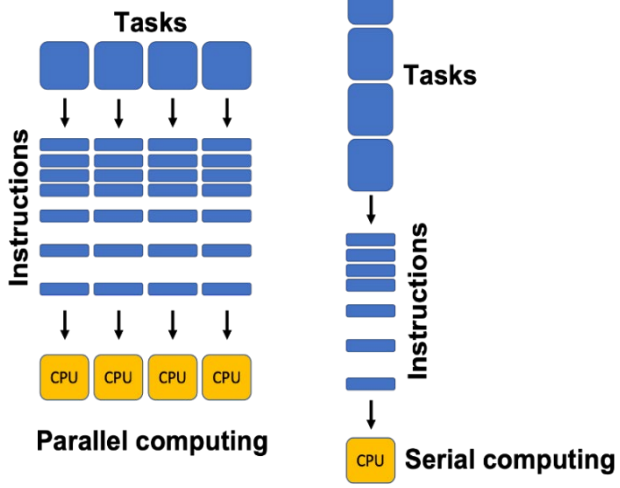
```
plt.figure(figsize = (8, 6))
plt.plot(t_e,y_I,  label= 'Rectangular current pulse')
plt.plot(sol_1.t, sol_1.y[2], label= 'Componet $m_z $ at G=8' )
plt.plot(sol_2.t, sol_2.y[2], label= 'Componet $m_z $ at G=%4.2f' %G)
plt.xlabel('t', size=16)
plt.ylabel('$m_z(t)$', size=16)
plt.legend(fontsize=12)
plt.show()
```




```
#plt.figure(figsize = (8, 6))
fig, ax1 = plt.subplots(figsize=(8, 8))
# mask out the negative and positve values, respectively
#Zpos = np.ma.masked_less(alpG[:,:,0], 0)
Z1 = Zc.reshape(N, N)
plt.imshow(Z1, interpolation='bilinear', cmap='Blues')
#plt.contourf(X, Y, Zc, 100)
#fig.colorbar(Zc, ax=ax1)
plt.show()
```



Parallel implementation with Python



Define a function called by each process

```
from joblib import Parallel, delayed
import numpy as np
```

```
def funk_parall(k):
    i=k%N
    j=k//N
    mz_sol=0
    G=G0+delta_G*i
    alpha=alpha0+delta_alpha*j
    f = partial(my_sfs, G=G, r=r, alpha=alpha, \
                As=As, t_s=t_s, delta_t=delta_t)
    t_e=np.linspace(0,60,1000)
    s0 = np.array([0, 0, 1, 0])
    sol_i=solve_ivp(f,[0,60],s0, t_eval=t_e) # method = 'Radau'
    if sol_i.y[2][999] < 0:
        mz_sol= -1
         # alpGxy[i+j*N,2] =- 1
    return mz_sol
```

Serial mode calculation

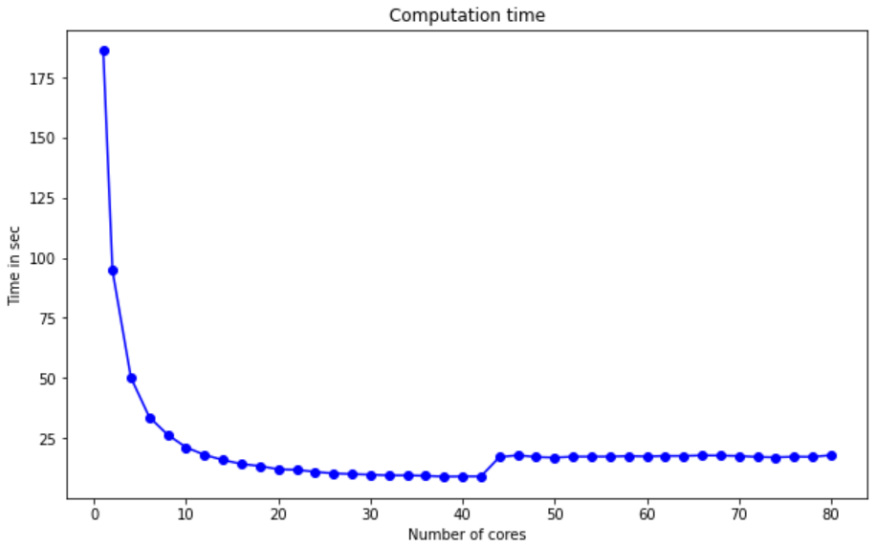
```
t0 = time.time()
rez= Parallel(n_jobs=1)\
    (delayed(funk_parall)(k) for k in range(N*N) )
t1 = time.time()
print(f'Execution time {t1 - t0} s')
```

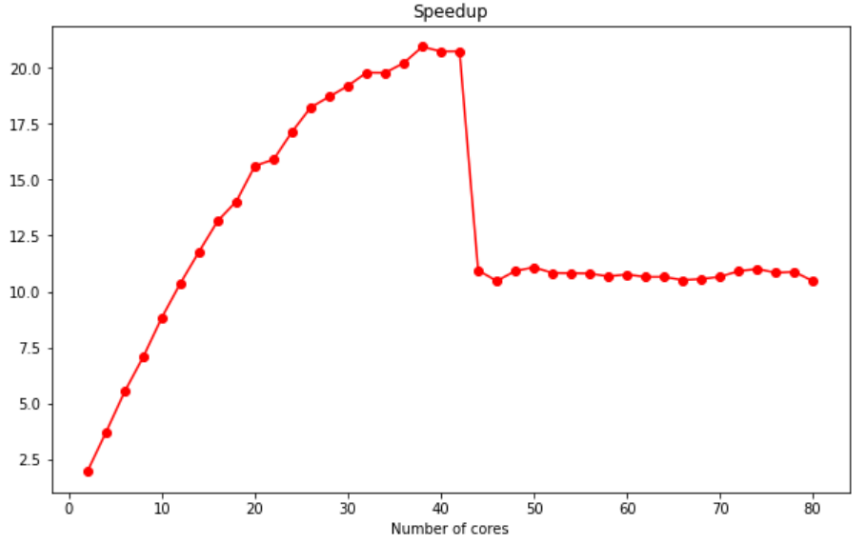
```
Execution time 159.9254457950592 s
```

Computing in Parallel Mode

```
t0 = time.time()
rez= Parallel(n_jobs=6)\
    (delayed(funk_parall)(k) for k in range(N*N) )
t1 = time.time()
print(f'Execution time {t1 - t0} s')
```

```
Execution time 34.51503801345825 s
```





Educational program on the MICC

Training courses, master classes and lectures

LIT staff and leading scientists from JINR and its Member States

Leading manufacturers of modern computing architectures and software

Parallel programming technologies

OpenMP

MPI

OpenCL

NVIDIA CUDA

Tools for debugging and profiling of parallel applications

Work with applied software packages

COMSOL MULTIPHYSICS

ROOT Data Analysis Framework

MATLAB

Wolfram Mathematica

GEANT4

Maple

Frameworks and tools for ML/DL tasks

jupyterhub

TensorFlow

scikit learn

NumPy

Thanks for your attention!

This work was supported by the Russian Science Foundation under grant No 22-71-10022

The XXVI International Scientific Conference of Young Scientists and Specialists (AYSS-2022)
Dubna, JINR, 24 – 28 October 2022