ML/DL/HPC ecosystem for solving applied tasks

O.I. Streltsova

On behalf of
Heterogeneous Computation Team HybriLIT
Laboratory of Information Technologies, JINR, Dubna, Russia

HYBRILIT WORKSHOP
2025

ML/DL/HPC Ecosystem of the HybriLIT Heterogeneous Platform

Component for educational purposes (without GPUs)

For teaching students
https://studhub1.jinr.ru

For conducting workshops within the framework of JINR scientific events
https://studhub2.jinr.ru
https://studhub3.jinr.ru

Component for carrying out resource-intensive computations (with GPUs)

https://jhub1.jinr.ru
https://jhub2.jinr.ru

ML/DL/HPC Ecosystem

HPC component for scientific projects (with installed specialized libraries)

BioProject services
https://cell.jinr.ru
http://mostlit.jinr.ru
http://bio-dashboards.jinr.ru/morris

Jupyter Books infrastructure
http://studhub.jinr.ru:8080/jjbook
http://studhub.jinr.ru:8080/books
http://studhub.jinr.ru:8080/itschool2024

CVAT services
http://159.93.36.88:8080
http://159.93.36.67:8080

A polygon for visualization of brain CT data
hlit-th-ct.jinr.ru

A polygon for quantum computing
https://ampere05.jinr.ru

Основные факторы роста применимости алгоритмов на базе ML/DL

Основные факторы роста применимости алгоритмов на базе ML/DL

ДАННЫЕ

Предобученные Модели

IMAGENET

http://www.image-net.org

PASCAL2

Pattern Analysis, Statistical Modelling and Computational Learning

https://deepai.org/dataset/pascal-voc

NETFLIX

Netflix Prize
(с 2006 года)

https://www.netflixprize.com/

Netflix Prize — открытое соревнование на лучший алгоритм предсказания оценки, которую зритель поставит фильму, на основе предыдущих оценок этого и других зрителей.

kaggle

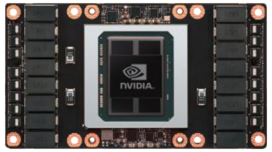
https://www.kaggle.com/

Kaggle — платформа (среда) для анализа данных, проведению соревнований, размещению наборов данных…..

Основные факторы роста применимости алгоритмов на базе ML/DL. Данные

Вычислительные архитектуры и IT-решения

NVIDIA Tesla V100, A 100, H 100



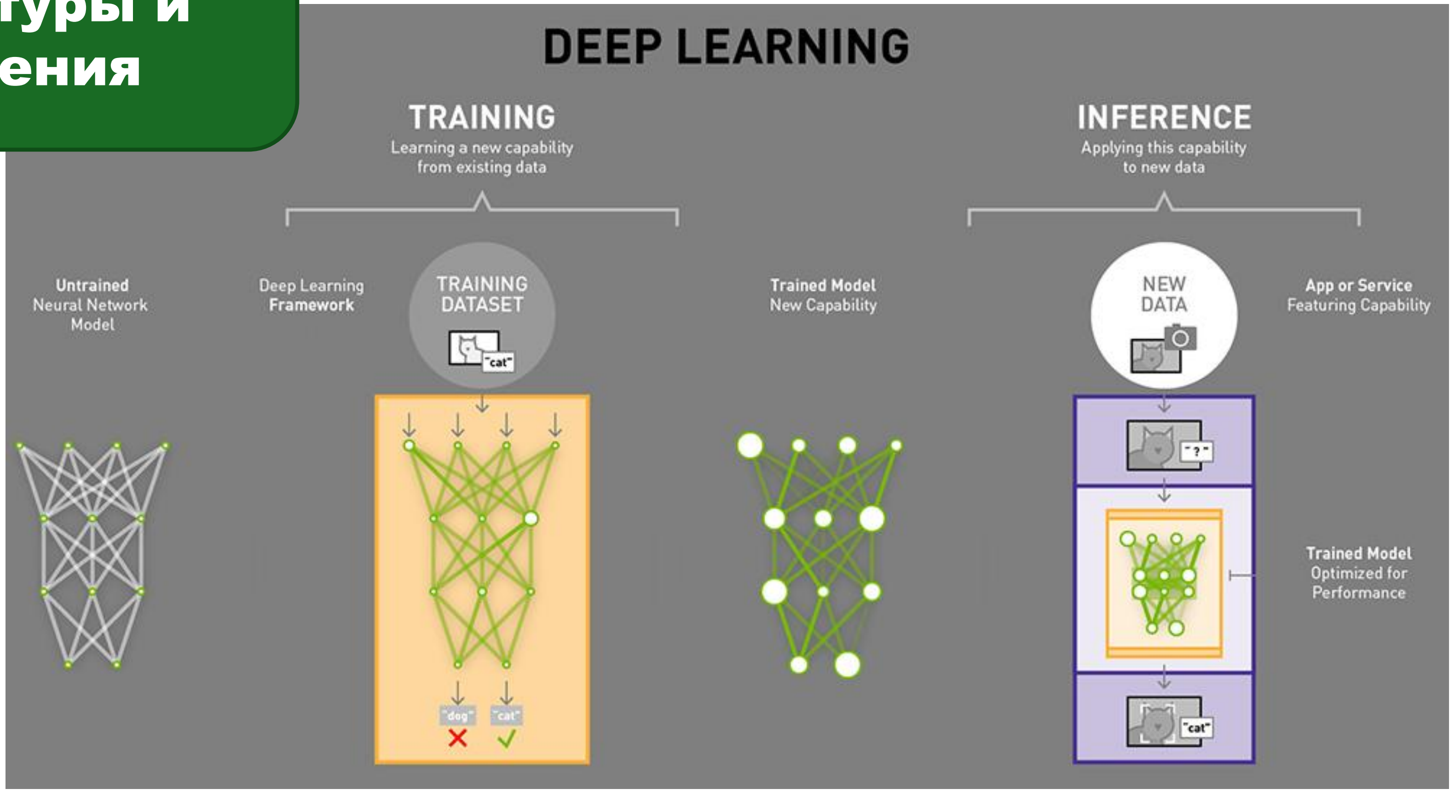


Fig. from https://blogs.nvidia.com/blog/2016/08/22/difference-deep-learning-training-inference-ai/

Основные факторы роста применимости алгоритмов на базе ML/DL

Алгоритмы ML/DL/AI

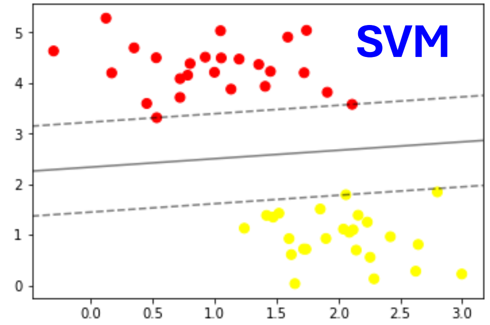
Библиотеки/фреймворки:

jupyter

K Caffe

TensorFlow PYTORCH theano

scikit learn NumPy



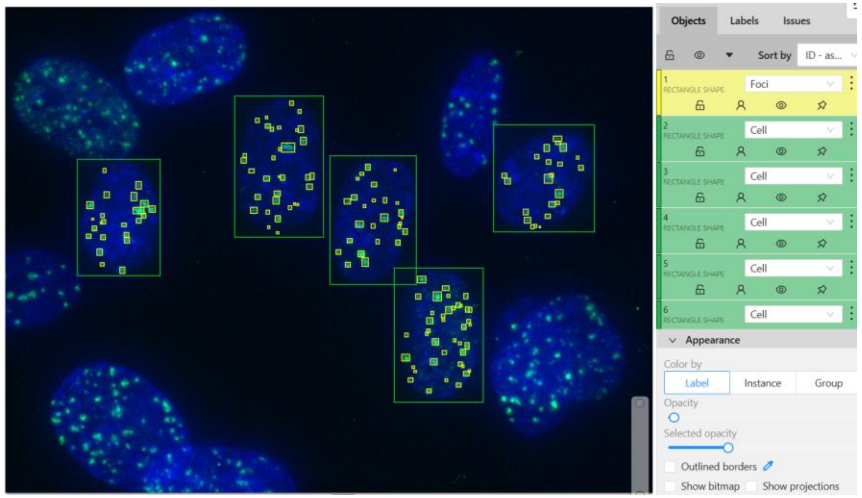
Neural style transfer

ML/DL/HPC Ecosystem of the HybriLIT Heterogeneous Platform

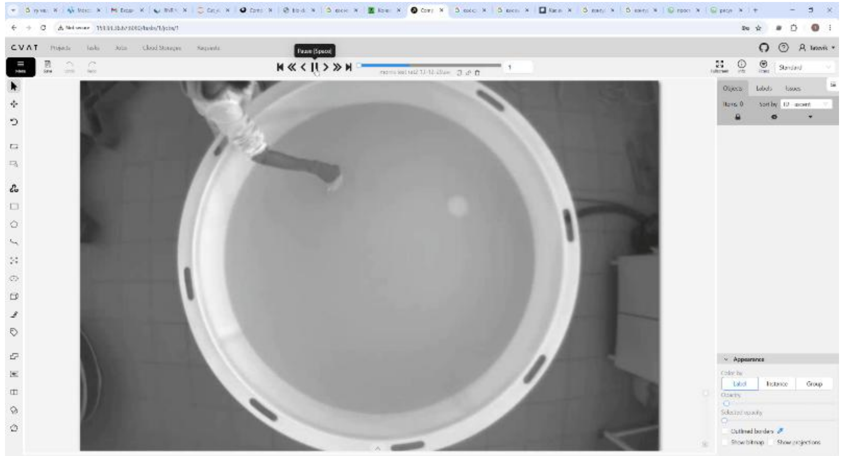
BioProject Services

MLIT, LRB, Dubna university, PFUR, TvSU

CVAT. Image & Video Data Annotation Platform

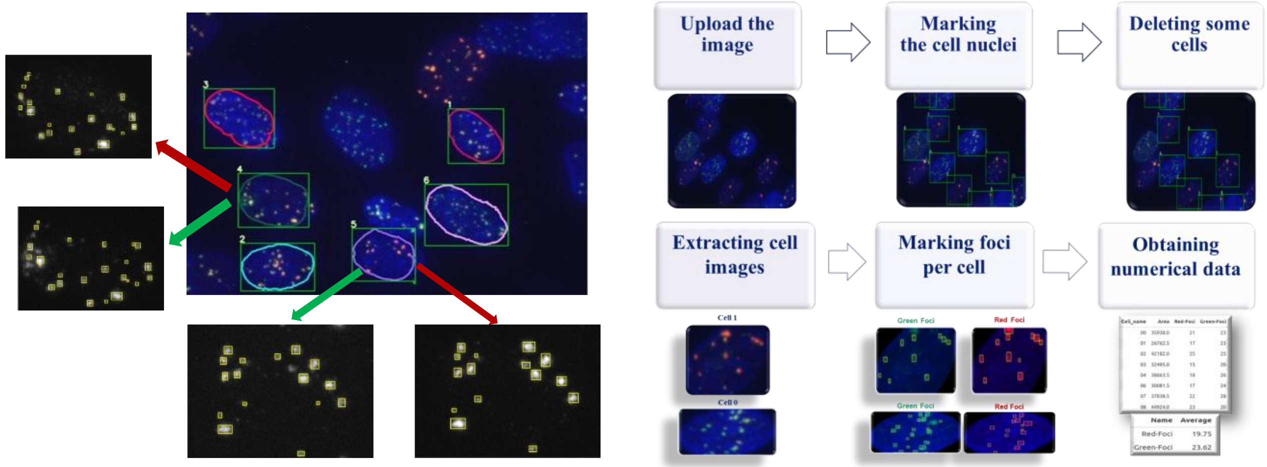


Annotation data: pictures and video

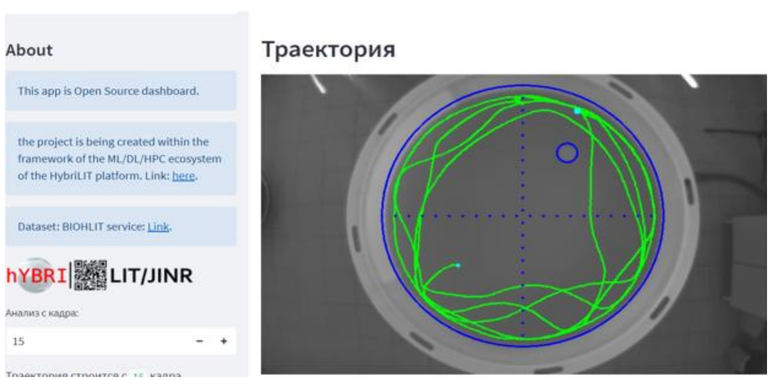


MOSTLIT. Service for FOCI detection and analysis

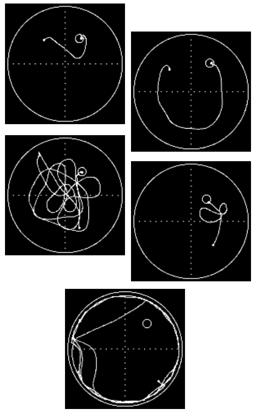
M. Zuev, O. Streltsova, S. Shadmehri, T. Bezhanyan



Web service for “Morris Water Maze” behavioral test



name	score
Total time traveled (sec)	61.0
Total distance (cm)	1635.39
FPS	30
Speed (cm/s)	71.49
First frame	26
Last frame	1856



Shadmehri, S., Bezhanyan, T., Bondarev, M. *et al.* A Deep Learning Model for Automated Quantification of DNA Repair Foci in Somatic Mammalian Cells. *Phys. Part. Nuclei* **56**, 1623-1627 (2025).

Bezhanyan T., Shadmehri S., Streltsova O. *et al.* Algorithm for the Analysis of the Laboratory Animal Trajectories in the “Morris Water Maze” and Its Implementation as a Web Service. *Phys. Part. Nuclei* **56**, 1370-1374 (2025).

hYBRI

CVAT. Image & Video Data Annotation Platform

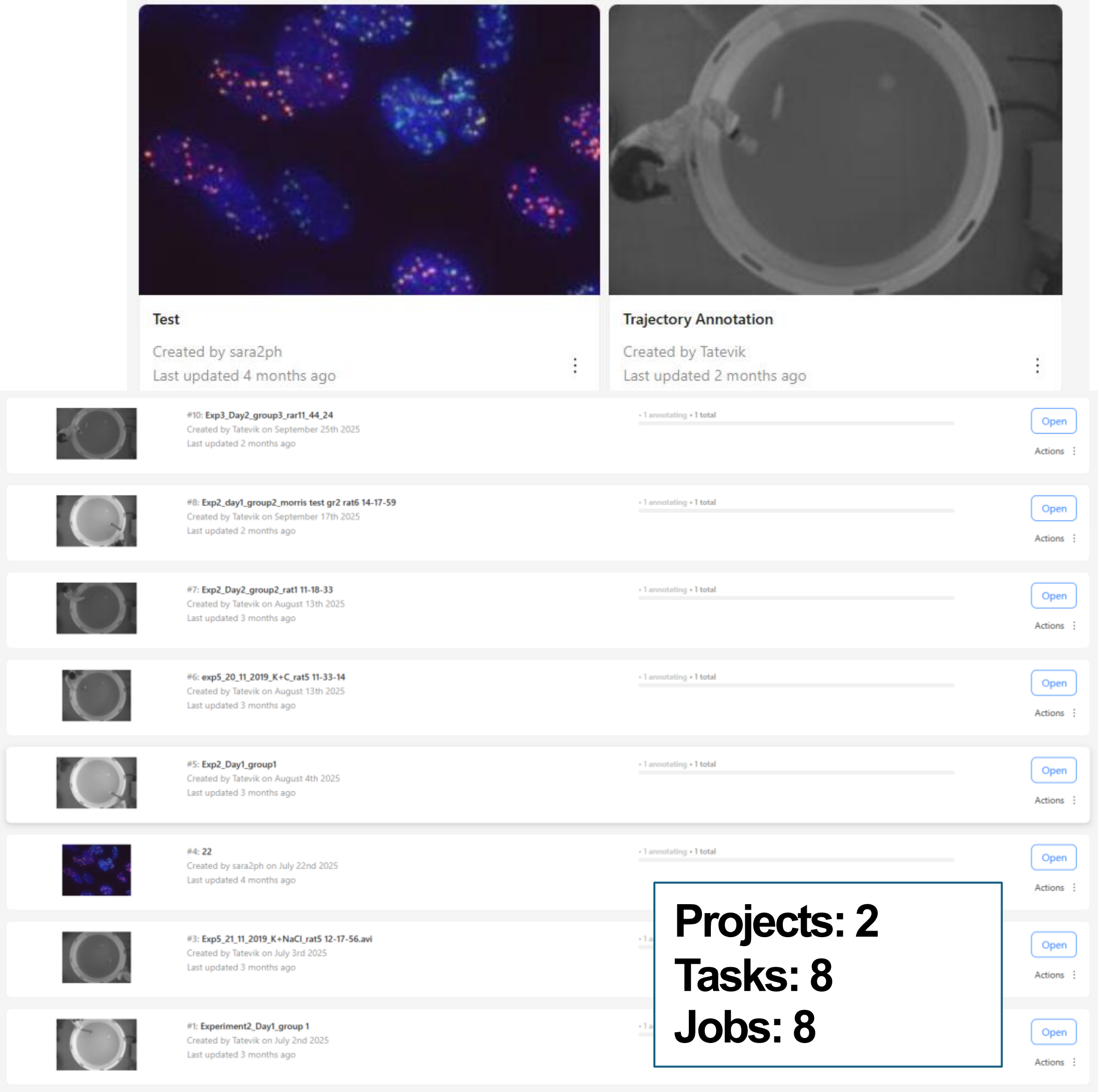
hYBRI

http://159.93.36.88:8080
CVAT version 2.2.0

CVAT

http://159.93.36.67:8080
CVAT version 2.40.0

Projects: 20
Tasks: 134
Jobs: 134



Projects: 2
Tasks: 8
Jobs: 8

YOLO11 neural network model for object detection

To improve the tracking algorithm for laboratory animals in the Morris Water Maze setup, the **YOLO11** neural network model was chosen as a pre-trained model for frame-by-frame object detection and the implementation of the "detection-based tracking" approach.

Data labeling was carried out on the **CVAT** platform deployed on the HybriLIT Platform (JINR LIT)

About **8000** frames from **7** video files were annotated using the CVAT (*Computer Vision Annotation Tool*) platform and used to train, validate, and test the YOLO11 model.

Модель YOLO11 (You Only Look Once): https://docs.ultralytics.com/ru/models/yolo11/

Research of systems based on Josephson junctions

MLIT, BLTP

Bashashin M., Kokaev D., Kulikov K., Nashaat M., Nechaevskiy A., Rahmonov I., Rahmonova A., Shukrinov Yu., Streltsova O., Zuev M.

http://hlit.jinr.ru/math-modeling-supercond-nanostructures-ru-2

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:

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

Acceleration of multiparameter computations

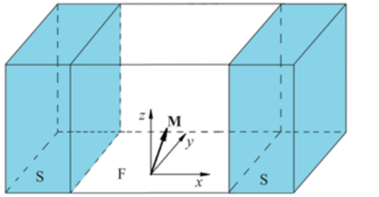
Joblib is a set of tools to provide lightweight pipelining in Python

Numba is an open-source JIT compiler that translates a subset of Python and NumPy code into fast machine code.

Calculation of the amplitude dependence of the Shapiro step width

We create a function to calculate the current-voltage characteristic and find the Shapiro step width

```
def Shapirostepsize(A, omega, n_harm, epsilon, Ilimit,
                    beta, deltaIext, ntmin, nt, deltat):
    a = 1
    t0 = 0
    Iext = 0
    Istep_list = []
    s0 = np.array([0, 0, 0])
    getstep = False  # Variable for checking whether a step is hit
    while Iext < Ilimit:
        res = cvcpoint(s0, Iext, beta, A, omega, nt, t0, ntmin, deltat)
        Vav = res[0]
        s0 = res[1]
        # Condition for reversing the direction of the current when first hitting a step
        # Required to fully obtain a step
        if ((a == -1) and (getstep == False) and np.abs(Vav-(n_harm*omega))
            a = 1
            getstep = True

        if ( (a==1) and (getstep==False) and np.abs(Vav-(n_harm*omega))<epsi
            getstep=True

        #Algorithm for determining the maximum and minimum current values wh
        if(np.abs(Vav-(n_harm*omega))<epsilon):
            Istep_list.append(Iext)
        Iext+=a * deltaIext
        #The condition for changing the direction of the current when exitin
        if(Vav>n_harm*omega+0.05):
            a=-1
        #Condition for stopping the current cycle
        if (Vav<n_harm*omega-0.05 and a==-1):
            break
        #End of current cycle

    #Calculation of step width
    Istep = max(Istep_list)-min(Istep_list)

    return Istep
```




```
t_start = time.time()

while Iext < Ilimit:
    res = cvcpoint(s0, Iext, beta, A, omega, nt, t0, ntmin, deltat)
    Vav = res[0]
    s0 = res[1]
    Vplot.append(Vav)
    Iplot.append(Iext)
    Iext += a * deltaIext
    if(Iext > Iext_max):
        a = - 1
    if ((Iext < 0) and (a == - 1)):
        break
t_finish = time.time()
print(f'Execution time {t_finish - t_start} s')
```

```
Execution time 46.573830127716064 s
```

Now we can plot the obtained current-voltage characteristics with and without external radiation

```
fig = plt.figure(figsize=(8, 6))
plt.plot(Iplotrad, Vplotrad, label='CVC with radiation', linewidth=3.0)
plt.plot(Iplot, Vplot, label='CVC without radiation', linewidth=3.0)
plt.xlabel('I', size=12)
plt.ylabel('V', size=12)
plt.legend(loc='upper left')
plt.show()
```

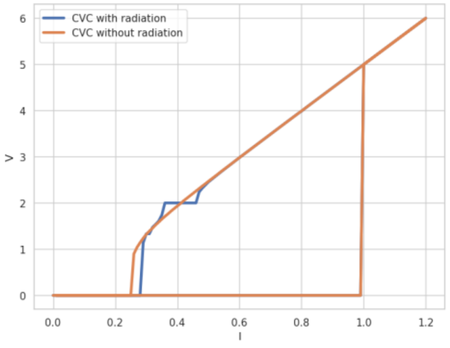


Fig. 4. Current-voltage characteristic with ($A = 0.5$) and without ($A = 0$) external radiation

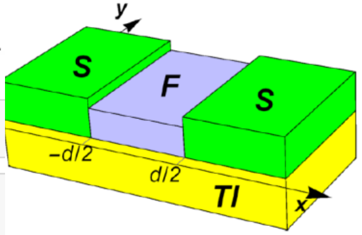
Jupyter Books for modelling superconductor/magnetic hybrid nanostructures

Bashashin M., Butenko Yu., Kokaev D., Kulikov K., Nechaevskiy A., Rahmonov I., Rahmonova A., Streltsova O., Zuev M.

http://studhub.jinr.ru:8080/jjbook, http://studhub.jinr.ru:8080/books, http://studhub.jinr.ru:8080/itschool2024

As part of a joint project between **MLIT** and **the Laboratory of Theoretical Physics (BLTP)** on modelling hybrid superconductor/magnetic nanostructures, a package of tools in the form of Jupyter Notebook, which are posted in the format of electronic publications **Jupyter Book** on the Platform resources has been developed.

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, alp
               As=As, t_s=t_s, delta
    t_e=np.linspace(0,60,1000)
    s0 = np.array([0, 0, 1, 0])
    sol_i=solve_ivp(f,[0,60],s0, t_ev
    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
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
t1 = time.time()
print(f'Execution time {t1 - t0} s')
```

```
Execution time 34.51503801345825 s
```

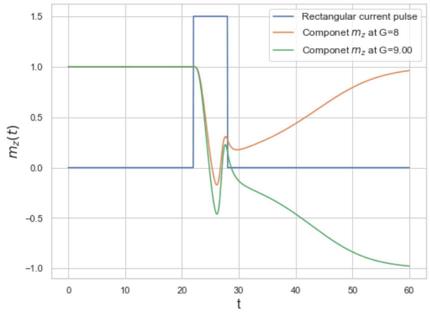
Calculations for different values of parameters

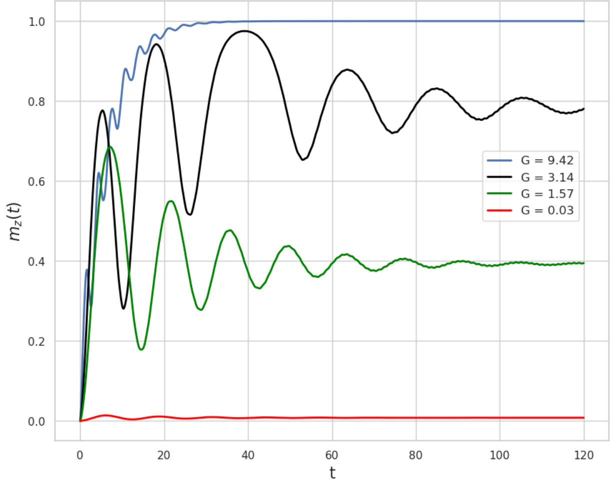
To analyze the possibility of reversing the magnetic moment of the ϕ_0-Josephson junction at different values of the parameters, we will carry out calculations for G=8.9.

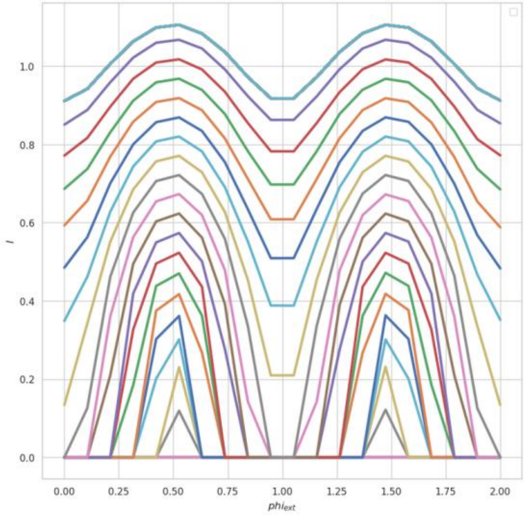
```
from scipy.integrate import solve_ivp
from functools import partial
```

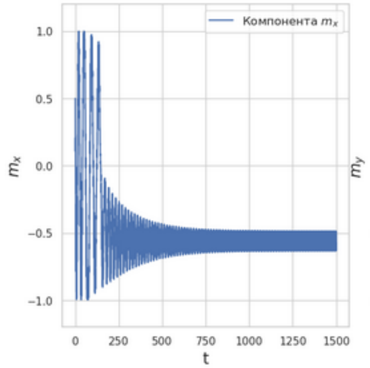
```
G=9
f = partial(my_sfs,  G=G, r=r, alpha=alpha, \
                As=As, t_s=t_s, delta_t=delta_t)
#t_e = np.arange(0, 25, 0.0001)
t_e=np.linspace(0,60,1000)

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

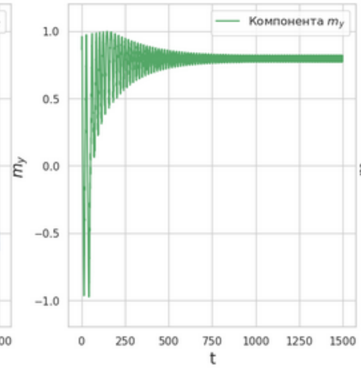
```
plt.figure(figsize = (8, 6))
plt.plot(t_e,y_I,  label= 'Rectangular current pulse')
plt.plot(sol_1.t, sol_1.y[2], label= 'Componet $m_z $ at G=8' )
plt.plot(sol_2.t, sol_2.y[2], label= 'Componet $m_z $ at G=%4.2f' %G)
plt.xlabel('t', size=16)
plt.ylabel('$m_z(t)$', size=16)
plt.legend(fontsize=12)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

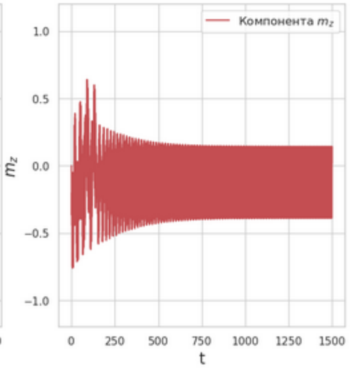












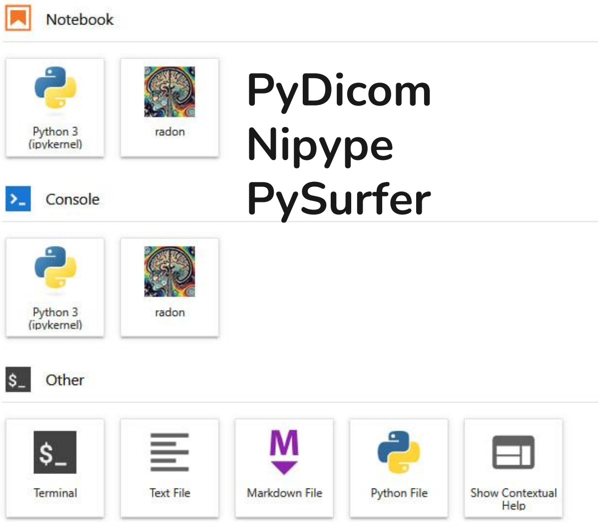
This work was supported by Russian Science Foundation grant № 22-71-10022.

hYBRI

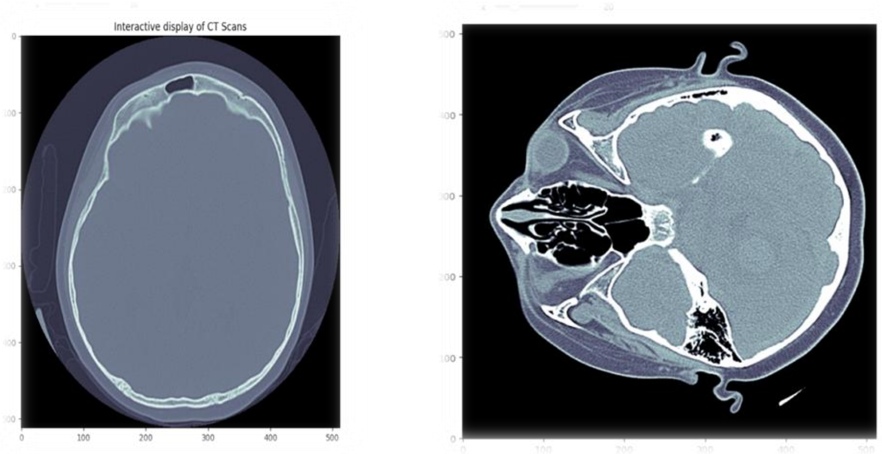
A polygon for visualization of brain CT and MRI/fMRI data

MLIT, BLTP, FMBA

JupyterLab interface

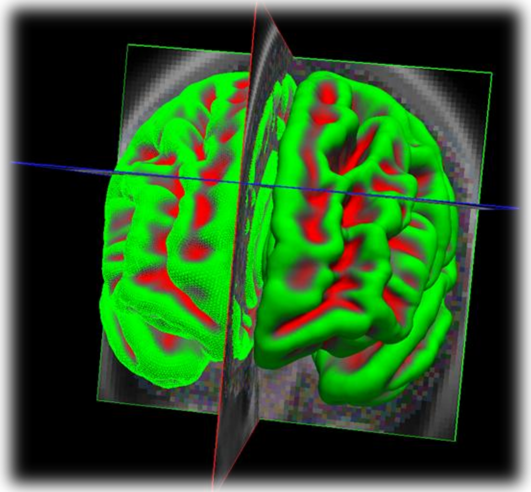


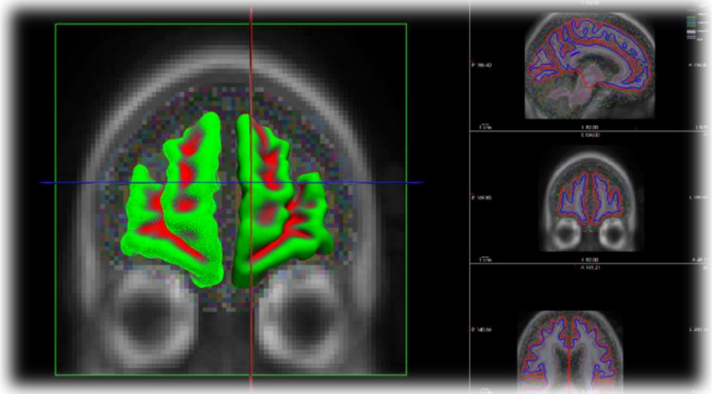
PyDicom
Nipype
PySurfer



SSH-access

- **MRI Convert** for converting the NIFTI file from the original set of DICOM files;
- **FreeSurfer** for NIFTI file processing;
- **FreeView** (viewer from the FreeSurfer package) 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





Anikina A., Andreev G., Andreev V., Anikin I., Belyackov D., Grigorov G., Kokorev A., Sharipov Sh., Streltsova O., Zuev M.
Polygon for visualization of brain computed tomography data. *Phys. Part. Nuclei* (sent to review).

hYBRI

Tutorial: Scientific Computing with Python on the HybriLIT ML/DL/HPC Platform Ecosystem

I. Алгоритмический путь

Algorithmically

Метод простой итерации

- Метод простой итерации с ускорением по Чебышёву
- Метод Якоби
- Метод Зейделя
- ……

II. JIT-компиляция для Python

JIT compilation for Python

Numba

Numba is a Python compiler from Anaconda that can compile Python code for execution on **CUDA-capable GPUs** or multicore CPUs.

III. Распараллеливание алгоритмов

Parallelization of algorithms

- Вычисления на CPU
- Вычисления с использованием ускорителей GPU NVIDIA …

Спасибо за внимание!