1 LABORATORY STATUS

The Nuclear Physics Methods (NPM) Laboratory was formally established in the beginning of 2019 based on the Nuclear Physics Methods group that was functioning in MIPT since 2016 and was backed by general physics department and later High Energy Physics Laboratory.

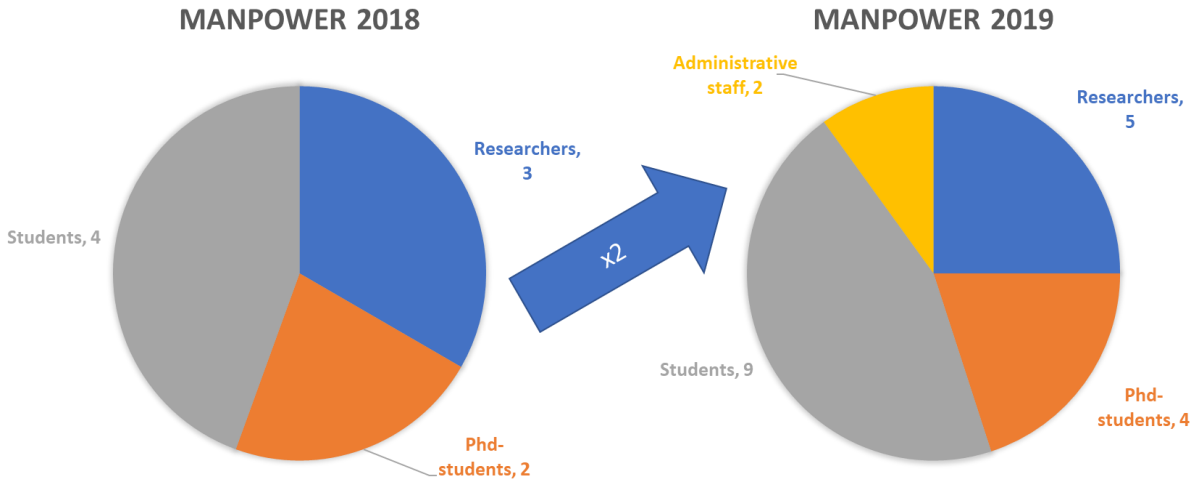

Figure 1. Manpower dynamics in the laboratory.

From the beginning, the NPM Laboratory was conceived as a mixed scientific and educational project with the intention of closing the gap between higher education and academic research. Currently, there are three main areas of work in the Laboratory:

1. The core field of study is non-accelerator physics, which nowadays is mostly represented by low-background deep underground experiments with some rare additions of very special setups on the surface. Non-accelerator experiments are usually smaller (both in size and in the number of people involved) compared to accelerator ones, yet they often employ unique experimental techniques as well as data analysis methods.
2. The second important branch of laboratory's activity is computer simulations in particle physics. This involves using standard packages like Geant4 as well as development of our own simulation software.
3. The quality of computer software is critical for particle physics, and we've seen room for improvement here for a long time. Luckily, we had few good programmers in our team - so we made a move to bring modern software development practices and standards to particle physics and, more importantly, work on teaching better programming to a new generation of physicists. In 2019 thanks to close ties with JetBrains and research grant from the JetBrains Research foundation, we can finally name scientific software as one of our primary research targets.

2 RESEARCH

2.1 NON-ACCELERATOR PARTICLE PHYSICS

2.1.1 Muon monitor

One of the key requirements for nuclear and particle physics experiments searching for rare processes is knowing the background conditions of underground experimental sites. To design an active 'veto' system of modern experiments and reduce the background from cosmic rays it is necessary to know not only the integral value of the residual muon flux but also its angular distribution. In 2013 in Canfranc Underground Laboratory a "MuonMonitor" experiment to measure an angular distribution of atmospheric muons flux was started. The NPM laboratory took part in processing of the experimental data. In 2019 an article with the first results was published in European Physical Journal C. Visualization of experimental events was written in Kotlin programming language.

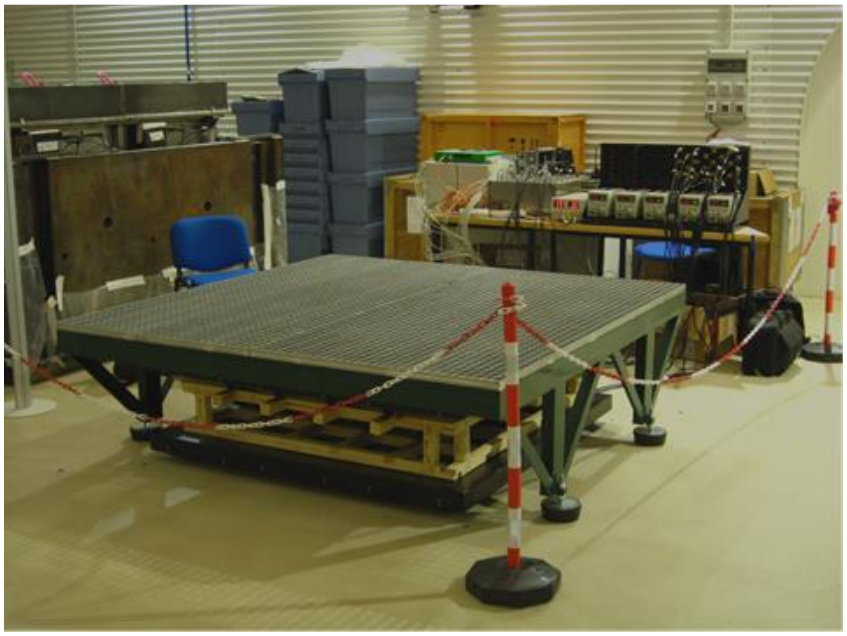

Figure 2. Muon monitor in assembled state alongside with shielding.

2.1.2 Troitsk nu-mass / TRISTAN / KATRIN

Figure 3 Troitsk nu-mass experiment layout.

Troitsk nu-mass is a relatively small experiment located in Troitsk. The experiment was started in 1985 and its initial goal was to search for the mass of the electron neutrino appearing in the process of Tritium beta-decay. The mass itself was not found (yet we are now reasonably sure that it is non-zero),

but the final result on upper boundary of electron antineutrino mass published in 2011 kept the world record until late 2019. The limit on parameters of light sterile neutrinos with mass around 1 keV (sterile neutrinos were the experiments' focus since 2012) is still the best in the world.

The whole data analysis and major part of the data acquisition system is written in Java and Kotlin.

In 2019 we were working on refining the analysis and signal processing techniques. Some data processing was rewritten in Kotlin (https://bitbucket.org/Kapot/lan10-processing/) from Python which brought significant increase in performance. The results are published in an article and Vasily Chernov is going to defend his PhD thesis at the beginning of 2020.

2.1.3 GERDA/LEGEND

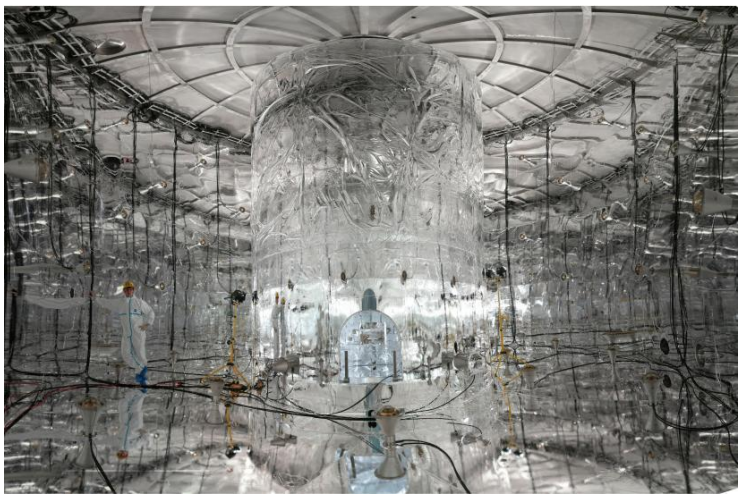

Figure 4. The inner part of GERDA's sensitive detector

GERDA (GERmanium Data Array) is an experiment designed to search for possible neutrinoless double-beta decay in Ge^{76} at the Laboratori Nazionali del Gran Sasso (LNGS). As a detector, the setup uses 40 kg of highly purified Ge^{76} placed in liquid argon supplemented by the water shield. So far, no signal of neutrinoless double-beta decay was found and a lower-limit for the half-life was deduced, equal to 2×10^{26} years at 90% CL. The next stage of search for neutrinoless double beta-decay is the LEGEND experiment. The mass of detector will reach one ton, and unless the signal is found, the lower limit for half-life is going to be improved up to 10^{28} years.

The laboratory is involved in the calculation of unremovable background component which arises because of capture of neutrinos from the sun. In 2019 the dedicated team finished most of calculations. The resulting background is negligible for GERDA and rather minor for LEGEND, but still important.

2.1.4 IAXO

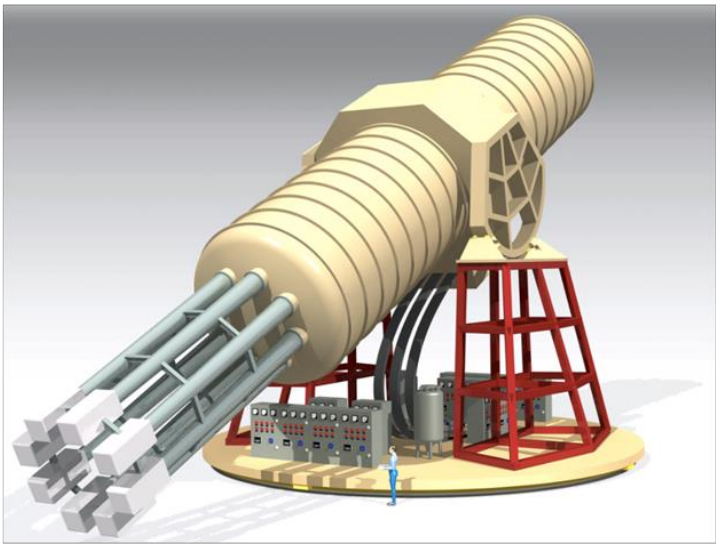

Figure 5. Expected IAXO layout (http://iaxo.web.cern.ch/)

IAXO (International AXion Observatory) is an experiment which plans to try detecting axion particles that are presumably constantly produced by the sun. The detection of axions is based on their decay into photons inside the magnetic field. While IAXO is rather small by the standards of particle physics, even its prototype (Baby-IAXO) development already unites almost a hundred of physicists.

The Laboratory joined this experiment in 2019. The contribution from NPM laboratory includes development of slow control system (we will work together with TANGO-controls team to modernize the system and introduce Kotlin wrapper for TANGO API) and visualization techniques.

2.1.5 Satellite detector

Earth orbit radiation safety requires a reliable and an accurate measurement of solar cosmic rays (SCR). A detector which is capable of real-time SCR monitoring with solar flare prediction is being developed. In addition, the detector can measure SCR spectrum during solar flares. Today this spectrum is poorly measured what makes the detector appealing for the sun research.

The detector is designed to measure electrons and protons with kinetic energies from 1 to 10 MeV and from 10 to 100 MeV respectively. The restoration of the particles' spectrum is carried out with the use of modern data analysis techniques including Turchin's regularization. Data analysis algorithms are implemented in a mix of different programming languages: Kotlin, Python, and Julia.

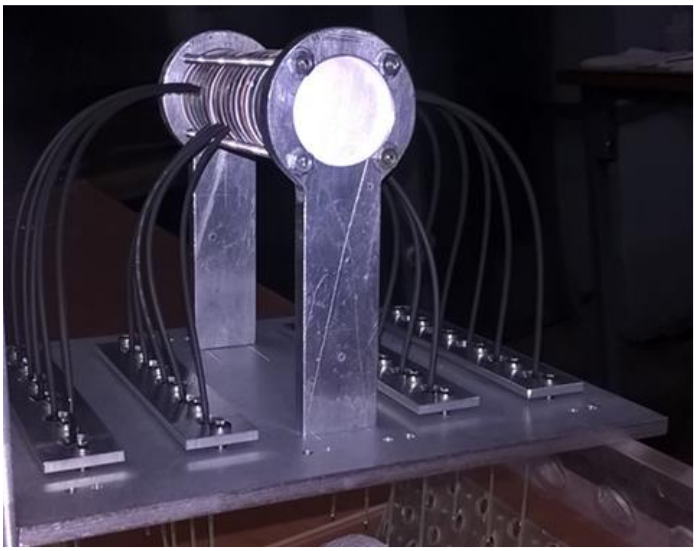

Figure 6. The prototype of a satellite detector.

2.2 SIMULATIONS

2.2.1 Gamma-rays for safety

It is important for national security to control the movement of dangerous or strategically important cargo such as explosives, radioactive materials, rare and precious metals. The fastest way to scan a container is to use X-ray source and appropriate detectors. It works not unlike medical X-ray scan, but with higher energies. The standard way to improve material recognition is to use twin-energies X-ray source, but some time ago there was a proposal (by the joint team from INR RAS and MIPT) to further improve the quality and reduce the costs by using energy sensitive detectors with single energy X-ray source.

The simulation was done with Geant4 toolkit with analysis being done using Python. Our team also developed and tested an algorithm to estimate basic chemical composition of materials inside the container. Results of this research were published in the Physics of Elementary Particles and Atomic Nuclei journal in 2019.

2.2.2 TGE (Gurevich / Dwyer)

Terrestrial gamma flash (TGF) and thunderstorm ground enhancement (TGE) phenomena are crucial for the understanding of atmosphere breakdown and physics of lightning generation. The initial theory was developed by Gurevich and included only runway breakdown description. It was later updated by Babich and Dwayer with the inclusion of positron and photon feedback mechanism, though even now it is obvious that those mechanisms could explain the whole set of observed phenomena. For example, simulation done by our team show that Dwyer mechanism is not able to provide stable feedback in observed atmospheric conditions. These results were obtained with Geant4 framework with some additional data processing modules in Python and Kotlin.

Modern TGF/TGE models with feedback use a rather simplified field structure and do not fully explain observed quantities. The reactor-like TGE (RL-TGE) model presented by our laboratory assumes more complicated stochastic field structure and considers not only one cell runway breakdown, but the whole global field cellular structure in the thundercloud. It allows to describe a wide variety of TGF-like events and potentially fills the gap in thundercloud parameters previously unaccounted by other

theories. Simulation code for this work was written in Kotlin (https://github.com/mipt-npm/rl-tge-sim). Results of the research were published in AIP Conference Proceedings in 2019.

2.2.3 Radio-signals from TGE

Relativistic runaway electron avalanches are considered to be the cause of TGE. Collisions during propagation through the air produce gamma-rays, using different mechanisms (Bremsstrahlung, for example). At the same time, accelerated charged particles always produce radiation according to Maxwell equations, so the goal is to estimate characteristics of this radiation, which seems to have radio frequency. The simulation for this work was done in Geant4 and data analysis is done in Python.

2.2.4 Spectator matter

While hot and dense matter is formed in the overlap zone of relativistic heavy-ion collision, residue of a nucleus remains relatively cold and forms what is called spectator matter. Spectator matter provides information about collision properties such as collision centrality, overlap zone initial shape, etc. Despite spectator matter has great scientific interest, there is no modern Monte-Carlo code which could simulate spectator matter precisely. Most Monte-Carlo codes used to simulate spectator matter are developed in Fortran 77/90 language and are not supported anymore.

Our laboratory is developing Abrasion-Ablation Monte-Carlo for Colliders (AAMCC) code based on abrasion-ablation mechanism using modern Glauber Monte-Carlo code for initial condition modeling and Geant4 toolkit for formed excited nuclei handling. AAMCC is written in C++14 and its architecture has modular structure unlike Fortran codes. Thereby one can substitute one model used in calculation with another which may be more suitable or connect AAMCC with some models that simulate overlap zone dynamics. Currently, AAMCC is being tested on a set of experimental data with aim to proof the correct work of fragment composition modeling. Very first results are published in the NUCLEUS-2019 conference proceedings.

2.3 Scientific software

2.3.1 DataForge

DataForge is a data processing framework initially used in Troitsk nu-mass experiment (https://bitbucket.org/Altavir/dataforge, https://bitbucket.org/Altavir/numass). It was using a mix of Java, Groovy and Kotlin but currently it is rewritten in Kotlin multiplatform (https://github.com/mipt-npm/dataforge-core). The framework uses several important concepts which allow to obtain, store and analyze data in a flexible and highly reproducible way, which in turn allows for easy parallel and distributed computing. The primary concept used in DataForge is metadata-processor (http://npm.mipt.ru/dataforge/docs.html#meta). It allows to define the process in a declarative way and perform configuration optimizations before starting actual process not unlike it is done in Gradle build system.

Some (somewhat outdated) documentation can be found at http://npm.mipt.ru/dataforge/.

2.3.2 Visualization systems

One of the primary projects developed in autumn 2019 was a framework for event visualization. The primary target is to create a stand-alone engine for visualization of events in particle physics experiments. Those events are usually displayed as a number of 3D geometric shapes and lines, but require convenient flexible constructor API, and multiplatform serialization capabilities (events could be produced in one platform and visualized in another platform).

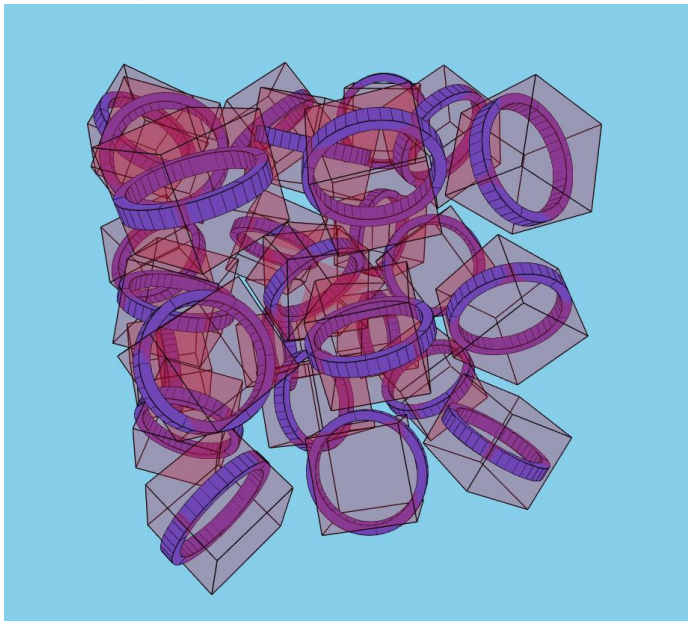

Figure 7. Demo picture for GEANT4 GDML geometry.

As part of prototyping, we published a Kotlin-multiplatform Plotly wrapper: https://github.com/mipt-npm/plotly.kt and a Medium article to accompany it: https://medium.com/@altavir/plotly-kt-a-dynamic-approach-to-visualization-in-kotlin-38e4feaf61f7.

Currently, the fully working prototype of visualization system written in Kotlin-multiplatform is located in this repository: https://github.com/mipt-npm/dataforge-vis/ (the name will be changed in future). The visualization system features DataForge-based core for (not only 3D) visualizable object tree with styling system, convenient builders and full reactive change tracking capacity. The 3D visualization system contains fully capable rendering using threejs library as well as (not yet fully capable) JavaFX back-end. In order to provide compatibility with existing scientific tools, we also devised an API to work with GDML geometry description language (https://github.com/mipt-npm/gdml.kt). The team is also currently working on interoperability with CERN ROOT framework. New visualization platform was presented on the BM@N experiment collaboration meeting and was preliminary accepted by software team. There are also plans to use it in other experiments.

2.3.3 Kmath

Kmath (https://github.com/mipt-npm/kmath) is a general-purpose multiplatform mathematical library written in Kotlin language. In Kmath we try to take a novel approach to mathematical library API design, utilizing mathematical abstractions to maximum and relying heavily on the context-oriented approach (for details, see https://proandroiddev.com/an-introduction-context-oriented-programming-in-kotlin-2e79d316b0a2). The distinct feature in terms of algebra operations is a separation of values and operations, meaning that the same object could feature different operations when used in a different lexical context (or scope).

Kmath goal is not to achieve the best performance, but to provide the most flexible and convenient API and then add existing platform-optimized libraries as bindings, so Kmath API could serve as a glue between them. Currently there are partial bindings for libraries like Apache commons-math, koma (https://github.com/kyonifer/koma) and JBR Viktor (https://github.com/JetBrains-Research/viktor).

2.3.4 BAT.jl

The BAT.jl (Bayesian Analysis Toolkit - https://github.com/bat/BAT.jl) is a Julia language package for distribution sampling and multivariate data analysis. BAT is one of the few international software-focused collaborations. Initially it was created to develop C++ version of the package (https://bat.mpp.mpg.de/), but later it became obvious that maintaining and developing something advanced in C++ is too tedious, so development was moved to Julia. Our laboratory task is to solve interoperability problem between Julia and other runtimes.

Also, in 2019 we have rewritten our Python implementation of Turchin's statistical regularization algorithm in Julia to make it more robust and allow easy interoperability with BAT. The algorithm allows to solve the Fredholm integral equation of the first kind with error propagation. For more details see https://github.com/mipt-npm/StatReg.jl).

3 EDUCATION

Most of laboratory senior members are actively working as teachers both at the university and in other organizations. Our main focus is general physics, but we are also active in other directions such as software development (especially Kotlin). In addition to basic courses, laboratory members regularly teach several special courses:

- Low-background subterranean experiments
- Statistical methods in experimental science
- An introduction into scientific computing in Kotlin language

As a laboratory we see two important educational missions for ourselves:

- Reduce the gap between academical researchers and higher education by providing a possibility for students to begin their scientific work early without hindering their studies.
- Bring modern programming best practices into science and raise a generation of programmer-physicists who can adopt those practices in actual research.

We are also actively working on modernizing general physics course in MIPT to include more computer-oriented stuff (automatic data acquisition, analysis and simulations).

4 PUBLICATIONS AND CONFERENCE TALKS

4.1 TALKS:

- ACAT-2019, Alexander Nozik, DataForge: declarative approach to scientific data processing automation. (https://indico.cern.ch/event/708041/contributions/3276140/)
- ACAT-2019, Alexander Nozik, Kotlin - new language for scientific programming. (https://indico.cern.ch/event/708041/contributions/3276141/)
- KotlinConf-2019, Kotlin for science, Alexander Nozik. (https://kotlinconf.com/talks/5-dec/114818)
- Nucleus-2019, Almaz Fazliakhmetov, Neutrino interaction with the Ga-Ge system and nuclear resonances
- Nucleus-2019, Grigory Koroteev, Decomposition of the charge-exchange strength functions for 76Ge and 74Ge isotopes
- AYSS-2019, Alexander Nozik, Kotlin language for science and kmath library. (https://indico.jinr.ru/event/756/session/4/contribution/316)
- AYSS-2019, Mikhail Zelenyi, Reactor like TGE model. (https://indico.jinr.ru/event/756/session/16/contribution/325)
- Particle and Cosmology 16th Baksan School on Astroparticle Physics, Egor Stadnichuk, Thundestorms and particle physics. (http://www.inr.ac.ru/~school/)
- MEDEX'19, Grigory Koroteev, (https://indico.utef.cvut.cz/indico/event/15/timetable/#20190531.detailed)
- TEPA 2019, Egor Stadnichuk, Completely random reactor model. (http://crd.yerphi.am/TEPA_2019)
- TEPA 2019, Timur Hamitov, Simulation of VHF signal from RREA. (http://crd.yerphi.am/TEPA_2019)

4.2 PUBLICATIONS:

Title	Authors	Journal
Cross Sections for Solar-Neutrino Capture by the 76Ge Nucleus and High-Lying Gamow—Teller Resonances	Vyborov, A.K., Inzhechik, L.V., Koroteev, G.A., Lutostansky, Y.S., Tikhonov, V.N., Fazliakhmetov, A.N.	Physics of Atomic Nuclei
Direct Search for keV-Sterile Neutrino in Nuclear Decay. Troitsk Nu-Mass (Scientific Summary)	Nozik, A.A., Pantuev, V.S.	JETP Letters
Shape-based event pileup separation in Troitsk nu-mass experiment	Chernov, V., Nozik, A.	Journal of Instrumentation
Reactor like TGE model	Zelenyi, M., Nozik, A., Stadnichuk, E.	AIP Conference Proceedings
Probing Majorana neutrinos with double-β decay	Gerda collaboration, включая Lev Inzhechik	Science
Kotlin language for science and Kmath library	Nozik, A.	AIP Conference Proceedings
Physics potential of the International Axion Observatory (IAXO)	IAXO collaboration, включая Alexander Nozik	Journal of Cosmology and Astroparticle Physics
Chemical Composition Analysis for X-Ray Transport Container Scans	Zelenaya, A., Zelenyi, M., Turinge, A.A., Nedorezov, V.G.	Physics of Particles and Nuclei
The Long-Term Stability of a Fused-Silica Proportional Counter	Abdurashitov, D.N., Chernov, V.G.	Instruments and Experimental Techniques
Statistical time analysis for regular events with high count rate	Nozik, A.	Journal of Instrumentation
Cross Section of Solar Neutrino Capture by 76Ge Nuclei	Vyborov, A.K., Inzhechik, L.V., Koroteev, G.A., Lutostansky, Y.S., Tikhonov, V.N., Fazliakhmetov, A.N.	Bulletin of the Russian Academy of Sciences: Physics
Cosmic-ray muon flux at Canfranc Underground Laboratory	Trzaska, W.H., et al.	European Physical Journal C
Designing Proton and Electron Detector for Monitoring Solar Cosmic Rays	Zelenyi, M.E., Stadnichuk, E.M., Nozik, A.A., Zimovec, I.V., Kudinov, A.G., Reznikov, I.K.	Bulletin of the Lebedev Physics Institute
Calculation of gain coefficient in Dwyer relativistic discharge feedback model of thunderstorm runway breakdown	Mikhail Zelenyi, Egor Stadnichuk and Alexander Nozik	EPJ Web Conf

5 PLANS

In the 2020 year, we will continue working on the projects described above. For software projects this includes adding documentation where it is needed.

We also plan to start two new software projects:

- SimBa is an experimental Monte-Carlo simulation engine which uses Kotlin coroutines and Bayesian network for optimized parallel simulation.
- We plan to cooperate with TANGO-controls collaboration in the development of simplified Java and Kotlin interfaces for slow control systems with subsequent bindings for existing software.

6 CONTACTS

Russian site: http://npm.mipt.ru/

JBR site: https://research.jetbrains.org/groups/npm

GitHub repositories: https://github.com/mipt-npm