LIPS - Learning Industrial Physical Simulation benchmark suite – Appendix

M. Leyli-Abadi, D. Danan, M. Yagoubi, S. Attoui
IRT SystemX, Palaiseau, France

A. Marot, J. Picault, B. Donnot
RTE France, Paris, France

P. Dimitrov, C. Etienam, A. Farjallah
Nvidia

Contents

A Datasheet for dataset for LIPS datasets   **3**

B Notation   **11**

C Benchmark Physical laws and variables   **12**
   C.1 Power grid use case 12
   C.2 Pneumatic use case 14

D Comparative related work table explained   **16**

E Logistics & available materials   **17**
   E.1 Datasets 17
   E.2 Getting started Notebooks 17
   E.3 Models 18
   E.4 Framework organisation 18
   E.5 Configuration Files 19
   E.6 LIPS usage 22
      E.6.1 Generating data 22
      E.6.2 Training an augmented simulator 23
      E.6.3 Evaluation using Benchmark class 24

F Experimental settings   **25**
   F.1 computing resources 25
   F.2 Augmented simulator visualization 25
   F.3 Preprocessing 25
   F.4 Hyper-parameter tuning 25

36th Conference on Neural Information Processing Systems (NeurIPS 2022) Track on Datasets and Benchmarks.

G Complementary results   **27**

    G.1 Numerical comparison table 27

    G.2 Physics compliances 27

H Accessibility   **31**

    H.1 Code repository and data availability 31

    H.2 Submission on Codabench 31

I Platform documentation   **33**

J Computation time investigation   **33**

    J.1 Physical solvers 33

    J.2 Augmented simulators 34

K Batch Size considerations for industrial Applications   **34**

L Further investigations (beside the provided developments)   **35**

    L.1 Convergence 35

    L.2 Investigation of required data volume for online learning 36

2

A Datasheet for dataset for LIPS datasets

Motivation

For what purpose was the dataset created? Was there a specific task in mind? Was there a specific gap that needed to be filled? Please provide a description.

The power flow distribution in a power grid is a cumbersome task for power grid operators as the number of factors influencing the grid and actions to be taken is humongous. To alleviate their task, the simulators are designed to evaluate the result of the set of possible actions on the basis of a current grid state. In this context, it is important to ensure the grid security during the various situations and while taking an remedial action. It is important that the developed simulators could compute the necessary grid state variables with high precision and in reasonable time. The datasets analyzed in this work are designed to simulate the real-world situations (e.g., line disconnections due to overheating), and to evaluate the performance of different simulators (e.g., physics-based or augmented simulators) tailored to different task and applications. This dataset also represents comprehensive distributions of grid states to unlock model training to enable comparison with approaches based on numerical solver or equation linearization.

Regarding the pneumatic case, designing and testing tires performances require access to critical information that are quite difficult to obtain, due to the physical problem inherent complexity and the time-consuming analysis that classically provide them. For this reason, in the context of practical industrial applications, those analysis are hardly used. However, with a developed simulators able to compute the tire states with a high precision within an acceptable time available, it would be possible to democratize its usage. The datasets considered here aim to represent the actual real-world configurations (support vehicle weight, rolling) in a representative way. They are also used for the model training of augmented simulators in order to compare their performances against approaches involving a classical numerical PDE solver.

Who created the dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)?

For power grid, the Benchmark datasets all depart from the same published datasets of realistic production and consumption distributions [1], [2] created by RTE and maintained by EPRI (Electric Power Research Institute) on Azure Servers. The benchmark datasets created by RTE AI Lab however each differs from the application specific grid topology variations which are applied using Grid2Op [3] framework. Methods in LIPS framework are provided to generate this same data. Ground truth of physical variables are further computed using LightSim2Grid [4] physical solver with industrial-like performance on the selected grids.

For pneumatic case, to the best of our knowledge, there is no such thing as published realistic datasets for tires that could have been used in this work. Therefore, the datasets is created by SystemX for physical configurations described for instance in [5], using the FEM physical solver Getfem [6]. Those datasets are generated using LIPS frameworks features, relying on this very solver.

Who funded the creation of the dataset? If there is an associated grant, please provide the name of the grantor and the grant name and number.

This dataset was funded as part of the HSA project (https://www.irt-systemx.fr/en/project/HSA/) - half funded by French National Research Agency - through a collaboration between IRT System x and RTE.

Any other comments?

Composition

What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)? Are there multiple types of instances (e.g., movies, users, and ratings; people and interactions be- tween them; nodes and edges)? Please provide a description.

3

Each instance represents one power grid state on a given situation at any hour, day, or month of a year. More specifically, it corresponds to production, consumption, grid topology (graph of electrical nodes and power lines) and power flow values over time.

For the pneumatic case, each instance represents a tire state computed at several points (nodes) within the tire geometrical description used by the physical solver (e.g. the mesh) on a given configuration.

How many instances are there in total (of each type, if appropriate)?

The number of instances can vary due to benchmark requirements, in particular when considering different system sizes. For each benchmark, we generate three different training datasets. A training dataset of size 150 000 is first provided on the environment IEEE 14 for benchmark 1. As there are 20 possible line disconnections and 7 reference topologies, this makes about 10 000 instances per line disconnection per reference topology on that grid. A larger dataset can readily be generated with the provided notebook. The datasets scale up according to the grid size, the number of topologies and line disconnection that are regarded and the number of energy mixes.

Dataset-Size= 10000*gridSizeFactor (number of representative situations) * N lines (number of disconnections) * T topo ref (number of reference topologies).

For the pneumatic case, for both benchmark, we also generate three datasets. The size of the training, testing and validation datasets are respectively 2100, 600 and 300, for benchark 1, and 280, 80 and 40, for benchmark 2.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? If the dataset is a sample, then what is the larger set? Is the sample representative of the larger set (e.g., geographic coverage)? If so, please describe how this representativeness was validated/verified. If it is not representative of the larger set, please describe why not (e.g., to cover a more diverse range of instances, because instances were withheld or unavailable).

Each generated dataset is designed to tackle a specific problem (risk assessment, solution search, etc.). From this point of view, the generated datasets could be considered as a sample of a more global dataset which can include all the possible situations and actions related to a power grid. Also note that the number of possible grid topologies can be almost infinite. 2^{16} is the order of 2 node configurations at a substation with 16 elements. Combinatorial complexity explodes even further when combining topology reconfiguration at different substations. The datasets are representative of the targeted tasks and human operator needs: they mostly work around a reasonable number of reference overall grid topologies, from which they can deviate by combining few unitary substation topologies.

For the pneumatic case, the generated datasets in both benchmarks could also be considered as a sample of a more global dataset which can include all the possible configurations. Let alone the fixed mesh considered in these datasets, where the domain meshes would be almost infinite otherwise, the fact that the sampling is done on a set of continuous interval for the inputs leads to a potentially infinite physical configurations. Likewise, the datasets are representative of the targeted tasks and driven by the real-word needs; they rely on a reasonable number of physical configurations from which one can derive a tire state fitted to one's need.

What data does each instance consist of? "Raw" data (e.g., unprocessed text or images) or features? In either case, please provide a description.

Each instance corresponds to one power grid state on a given situation. Hence, it includes the consumptions and productions at each substation and also the power flow through the power lines. More specifically, the voltages (Kv), currents (Amp), active (Mw) and reactive (Mvar) power values are provided. Those are "raw" data instances. Note that one representation of grid topology is given, but it could be represented in many different ways that could prove more or less successful.

For the pneumatic case, each instance corresponds to one tire state. Thus it includes, depending on the benchmark, the displacement field (mm) evaluated on each nodes of the physical domain and the contact stress field (MPa), evaluated on each nodes of the associated boundary within the physical domain. Those are also what would describe as "raw" data instances.

Is there a label or target associated with each instance? If so, please provide a description.

4

The objective of a simulator (physical or augmented) is to infer the power flow through power lines from the productions and consumptions. Hence, the problem could be expressed as a regression problem where the target corresponds to power flow variables. A physical simulator provides that target variables after computation. They can hence be used if training an emulator. But they could also be discarded if rather training a solver directly minimizing some Energy loss.

For the pneumatic case, the exact same logic still stands by replacing power flow prediction by tire state prediction.

Is any information missing from individual instances? If so, please provide a description, explaining why this information is missing (e.g., because it was unavailable). This does not include intentionally removed information, but might include, e.g., redacted text.

As we have used a physical simulator to generate the required datasets, there was not any missing information and we provide complete states.

Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)? If so, please describe how these relationships are made explicit. Instances can belong to a same grid operation scenario, e.g. a same 5 minute resolution monthly time series. But instances are sorted randomly in the end in the dataset and no explicit time dependency relationship are given and none is needed for the task presented.

There are no relationship between individual instances for the pneumatic case.

Are there recommended data splits (e.g., training, development/validation, testing)? If so, please provide a description of these splits, explaining the rationale behind them.

The proposed framework for benchmarking allows to generate datasets with variable size for any purpose (training, validation and test). It could also be the subject of a benchmark to infer the data volume required for training a model in order to obtain a desired performance. As an recommendation, the training dataset should be big enough to include various situations that a power network can handle during its lifespan (2x or 3x bigger than the test set). Validation and test sets are also provided and include out-of-distribution instances as specified in the proposal.

Are there any errors, sources of noise, or redundancies in the dataset? If so, please provide a description

There are no errors, sources of noise or redundancies in provided datasets.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? If it links to or relies on external resources, a) are there guarantees that they will exist, and remain constant, over time; b) are there official archival versions of the complete dataset (i.e., including the external resources as they existed at the time the dataset was created); c) are there any restrictions (e.g., licenses, fees) associated with any of the external resources that might apply to a future user? Please provide descriptions of all external resources and any restrictions associated with them, as well as links or other access points, as appropriate.

The provided datasets are self-contained and will remain constant. However, more datasets could be generated using the proposed benchmarking platform. The data generation part of our platform can be procedurally augmented through the open-source power grid platform called Grid2Op which is under Mozila Public License : https://github.com/rte-france/Grid2Op/blob/master/LICENSE.md

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor-patient confidentiality, data that includes the content of individuals' non-public communications)? If so, please provide a description.

No, as this is synthetic data. This is one main reason for providing synthetic data to avoid some existing confidentiality issues with real data for these critical systems.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? If so, please describe why.

No

Does the dataset relate to people? If not, you may skip the remaining questions in this section.

5

No

Does the dataset identify any subpopulations (e.g., by age, gender)? If so, please describe how these subpopulations are identified and provide a description of their respective distributions within the dataset.

No

Is it possible to identify individuals (i.e., one or more natural persons), either directly or indirectly (i.e., in combination with other data) from the dataset? If so, please describe how.

No

Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals racial or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security numbers; criminal history)? If so, please provide a description.

No

Any other comments?

Collection Process

How was the data associated with each instance acquired? Was the data directly observable (e.g., raw text, movie ratings), reported by subjects (e.g., survey responses), or indirectly inferred/derived from other data (e.g., part-of-speech tags, model-based guesses for age or language)? If data was reported by subjects or indirectly inferred/derived from other data, was the data validated/verified? If so, please describe how.

Each instance in our dataset represents a synthetic observation of a specific power grid state. There are the electrical measures expressing the power flow in a power network and directly observable.

In the pneumatic case, each instance represents a synthetic observation of a specific tire state, that is to say the mechanical measures related to a tire and directly observable.

What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)? How were these mechanisms or procedures validated?

Realistic timeseries for production and consumption localized over a grid were first generated through Chronix2grid package in published datasets [1], [2]. Then a simulator (software program running as a backend of Grid2Op platform) comes at the core of our proposed benchmarking framework to generate the data. However, the proposed LIPS benchmark platform extends this simulator and makes an API to facilitate the data generation for various benchmarking purposes and possible other industrial domains relying on other simulators.

As a matter of fact, it was possible to extend this framework for the pneumatic case, a completly different industrial domain, in order to generate data using the simulator Getfem [6], a FEM solver.

If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)?

The proposed framework allows to generate the synthesized observations with various sampling strategies. It could be probabilistic or deterministic as desired and the sampling probability is a parameter that can be adjusted by user. As an example, one can generate the power flow data by introducing some changes (line disconnection) with a desired probability p. As such, the generated dataset shall contain the changes or not with respect to the indicated probability. See dataset size rationale indicated described before when considering the number of instances.

Regarding the pneumatic usecase, for benchmark 1 a Latin Hypercube Sampling (LHS), a near-random sampler of parameter values arising from a multidimensional distribution, was used. For benchmark 2, we consider a simple uniform sampling.

Who was involved in the data collection process (e.g., students, crowd workers, contractors) and how were they compensated (e.g., how much were crowd workers paid)?

6

The data collection process is done using a computer software (power grid simulator developed by RTE France, FEM solver for the tire simulator). However, the provided datasets and corresponding designed scenarios are issued from the proposed LIPS platform. The authors created them as part of their full time job.

Over what timeframe was the data collected? Does this timeframe match the creation timeframe of the data associated with the instances (e.g., recent crawl of old news articles)? If not, please describe the timeframe in which the data associated with the instances was created.

The final benchmark datasets are generated and collected during june 2022. This is the results of several iterations and benchmark adjustements since summer 2021.

Were any ethical review processes conducted (e.g., by an institutional review board)? If so, please provide a description of these review processes, including the outcomes, as well as a link or other access point to any supporting documentation.

N/A

Does the dataset relate to people? If not, you may skip the remainder of the questions in this section.

No

Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (e.g., websites)?

No

Were the individuals in question notified about the data collection? If so, please describe (or show with screenshots or other information) how notice was provided, and provide a link or other access point to, or otherwise reproduce, the exact language of the notification itself.

N/A

Did the individuals in question consent to the collection and use of their data? If so, please describe (or show with screenshots or other information) how consent was requested and provided, and provide a link or other access point to, or otherwise reproduce, the exact language to which the individuals consented.

N/A

If consent was obtained, were the consenting individuals provided with a mechanism to revoke their consent in the future or for certain uses? If so, please provide a description, as well as a link or other access point to the mechanism (if appropriate).

N/A

Has an analysis of the potential impact of the dataset and its use on data subjects (e.g., a data protection impact analysis)been conducted? If so, please provide a description of this analysis, including the outcomes, as well as a link or other access point to any supporting documentation.

N/A

Any other comments?

Preprocessing/cleaning/labeling

Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of miss- ing values)? If so, please provide a description. If not, you may skip the remainder of the questions in this section.

No specific preprocessing is done in the raw datasets beside removing divergent simulation instances. However for training models, some functions are provided to run possibly useful preprocessing steps. Current preprocessing step allows to encode the network topology in different ways. Standardization of injections and power flows are also available (zero mean and unit variance).

7

Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)? If so, please pro- vide a link or other access point to the "raw" data.

The raw data is readily part of the datasets and no preprocessing is done for the provided datasets.

Is the software used to preprocess/clean/label the instances avail- able? If so, please provide a link or other access point.

N/A

Any other comments?

Uses

Has the dataset been used for any tasks already? If so, please provide a description.

The datasets from which the benchmark dataset is created has already been used for "Learning to run a power network challenge" competitions and internal researches on power grids. Topological change scope are explicitly defined here to fit the benchmark needs in addition and clearly specified in LIPS config files.

Is there a repository that links to any or all papers or systems that use the dataset? If so, please provide a link or other access point.

What (other) tasks could the dataset be used for?

In the present article, the dataset has been used to benchmark various surrogate models for power flow prediction. The dataset could be used for further research domains concerning power grids, such as network reliability, decision making, optimization of agent actions, etc. The dataset could also be used for the classification of blackout situations during the lifetime of a power grid. It requires some adaptations and is one of our insights for future works.

For the pneumatic case, the dataset has been used to benchmark various surrogate models for tire state prediction and, as such, could to used to investigate the performances of other various surrogate models.

Is there anything about the composition of the dataset or the way it was collected and prepro- cessed/cleaned/labeled that might im- pact future uses? For example, is there anything that a future user might need to know to avoid uses that could result in unfair treatment of individuals or groups (e.g., stereotyping, quality of service issues) or other undesirable harms (e.g., financial harms, legal risks) If so, please provide a description. Is there anything a future user could do to mitigate these undesirable harms?

Divergent simulations are removed from now but could be of interest later on to classify stable power grid states. However, we give the flexibility for users to re-run all simulations from initial environment and to keep these divergent simulations to form an extended dataset. This point is also one of our insights for future work.

Are there tasks for which the dataset should not be used? If so, please provide a description.

The current form of the dataset could be used for regression problems. One can infer the power flow from injections to the network. The users are free to select between the inputs and outputs of the model. In order to do a classification task, the dataset require more processing steps as it is mentioned in the previous answer.

Any other comments?

Distribution

Will the dataset be distributed to third parties outside of the en- tity (e.g., company, institution, organization) on behalf of which the dataset was created? If so, please provide a description.

The datasets is available publicly on Github : `https://github.com/IRT-SystemX/LIPS`

How will the dataset will be distributed (e.g., tarball on website, API, GitHub)? Does the dataset have a digital object identifier (DOI)?

8

The datasets used for experimentation will be available via Github link https://github.com/Mleyliabadi/LIPS. These datasets could also be reproduced via the developed platform and using the same seeds. The LIPS platform gives also the flexibility to generate further distributions envisaging other experiments.

When will the dataset be distributed?

The datasets will be distributed as soon as the benchmark is accepted.

Will the dataset be distributed under a copyright or other intel- lectual property (IP) license, and/or under applicable terms of use (ToU)? If so, please describe this license and/or ToU, and provide a link or other access point to, or otherwise reproduce, any relevant licensing terms or ToU, as well as any fees associated with these restrictions.

The datasets are generated using an open source platform called Grid2Op that is specialized on power grids and which is the property of RTE France. The Mozilla Public License applies and the license terms could be visited at https://www.mozilla.org/en-US/MPL/2.0/

Have any third parties imposed IP-based or other restrictions on the data associated with the instances? If so, please describe these restrictions, and provide a link or other access point to, or otherwise reproduce, any relevant licensing terms, as well as any fees associated with these restrictions.

No

Do any export controls or other regulatory restrictions apply to the dataset or to individual instances? If so, please describe these restrictions, and provide a link or other access point to, or otherwise reproduce, any supporting documentation.

No

Any other comments?

Maintenance

Who is supporting/hosting/maintaining the dataset?

The authors of the paper will be responsible to support the datasets.

How can the owner/curator/manager of the dataset be contacted (e.g., email address)?

The preferred way to contact the maintainers is to raise issues on our github page https://github.com/IRT-SystemX/LIPS. For the emergency cases, the authors of the paper could be contacted by mail l2rpn@chalearn.org.

Is there an erratum? If so, please provide a link or other access point

Any update, supplementary materials or information can be accessed via the github page of the platform https://github.com/IRT-SystemX/LIPS .

Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)? If so, please describe how often, by whom, and how updates will be communicated to users (e.g., mailing list, GitHub)?

As the datasets are synthesized and are self-contained, there will be no modifications in future. However, more datasets and use cases may be designed in future to address our insights and they will be shared via the platfom's github page.

If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., were individuals in question told that their data would be retained for a fixed period of time and then deleted)? If so, please describe these limits and explain how they will be enforced.

N/A

Will older versions of the dataset continue to be supported/hosted/maintained? If so, please describe how. If not, please describe how its obsolescence will be communicated to users.

9

As the dataset are synthesized and are designed to respond to a specific requirement, they will not present any modifications in future and we will keep them available. In the case of evolving requirements and objectives, new datasets and scenarios will be generated independently and will be included in github page.

If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so? If so, please provide a description. Will these contributions be validated/verified? If so, please describe how. If not, why not? Is there a process for com- municating/distributing these contributions to other users? If so, please provide a description.

Users are allowed to extend or augment the already generated datasets for their purposes. The proposed platform allows also the users to generate new datasets with ease and with required distribution and volumes, as well as new tasks to be benchmarked over. It is anticipated to incluse datasets comming from other industrial domains.

Any other comments?

10

B Notation

The notations alongside the description of the symbols used throughout the article for power grid and pneumatic use cases are shown in the following tables respectively.

NOTATIONS for power grid usecase

i	index associated with an individual (entity)
t	index associated with a time instant
k	index associated with a grid node
ℓ	index associated with a power line
or	index associated with the "origin" extremity of a power line by convention
ex	index associated with the second extremity of a power line by convention
j	the imaginary part of complex numbers
p_k	active power at node k
q_k	reactive power at node k
v_k	voltage at node k
θ_k	voltage phase (angle) at node k
p^l	active power flow over line l
q^l	reactive power flow over line l
a^l	current over line l
L	number of power lines in the network
K	number of substations in the network
N	number of samples in training set
M	number of samples in test set

NOTATIONS for pneumatic use case

Ω	the domain
Γ	the domain boundary subdivided into 3 parts Γ_1, Γ_2, Γ_3
$\boldsymbol{X}$	the initial position
$\boldsymbol{u}_\Omega$	the displacement field
ν, τ	are respectively the unit normal and tangential vector on Γ
$u_{\Omega\nu}, \boldsymbol{u}_{\Omega\tau}$	the normal and tangential component of $\boldsymbol{u}_\Omega$ on Γ
$\boldsymbol{\lambda}$	the contact stress field
$\phi(X) = \boldsymbol{X} + \boldsymbol{u}_\Omega$	the deformation
$\mathbf{F} = \nabla\phi = \mathbf{I} + \nabla\boldsymbol{u}_\Omega$	the deformation gradient
$\mathbf{C} = \mathbf{F}^T\mathbf{F}$	the Cauchy-Green tensor
$\mathbf{E} = \dfrac{1}{2}(\mathbf{C} - \mathbf{I})$	the Green-Lagrange tensor (deformation tensor)
$i_1(\mathbf{E}) = tr(\mathbf{E})$	first invariant of the deformation tensor
$i_2(\mathbf{E}) = \dfrac{(tr(\mathbf{E}))^2 - tr(\mathbf{E}^2)}{2}$	second invariant of the deformation tensor
$i_3(\mathbf{E}) = det(\mathbf{E})$	third invariant of the deformation tensor
W	the internal hyperelastic energy density
$\boldsymbol{\Pi}$	the first Piola-Kirchhoff stress tensor
$\Pi_\nu = (\boldsymbol{\Pi}\boldsymbol{\nu}) \cdot \boldsymbol{\nu}$	the normal stress on Γ
$\boldsymbol{\Pi}_\tau = \boldsymbol{\Pi}\boldsymbol{\nu} - \Pi_\nu\boldsymbol{\nu}$	the tangential stress on Γ
$\boldsymbol{\sigma} = \dfrac{1}{det(\mathbf{F})}\boldsymbol{\Pi}\mathbf{F}^T$	the Cauchy stress tensor
$\boldsymbol{\sigma}^{\text{dev}} = \boldsymbol{\sigma} - \frac{tr(\boldsymbol{\sigma})}{3}\mathbf{I},$	the deviatoric tensor of the cauchy stress tensor (where tr and det are respectively the classical trace and determinant operators for matrices.)

11

C Benchmark Physical laws and variables

We here list the physical variables and physical law verification considered through each benchmark.

C.1 Power grid use case

Input and output physical variables: We here display a table of input and output variables of the model expected for each benchmark, with related units and a brief description about them.

Table 1: Input and output variables of the model. Note that nodal variables are rather predicted at each line extremity in practice to keep output dimensions constant. ood subscript in benchmark highlights the concerned input variables with changing distribution in the related ood test set

Variable	Unit	Role	Description	Bench	IEEE14	IEEE39
p_{prod}	Mw	inputs	productions active power	1,2	6	22
v_{prod}	Kv		productions voltage	1,2	6	22
p_{load}	Mw		loads active power	1,2	11	37
q_{load}	MVar		loads reactive power	1,2	11	37
τ	Binary		topology of the power network	$1,2_{ood}$	80	236
L	Binary		line status of power network	$1_{ood},2$	20	59
a_{or}^{ℓ}	Amp	outputs	current at the origin side of a power line	1,2	20	59
a_{ex}^{ℓ}	Amp		current at the extremity side of a power line	1,2	20	59
p_{or}^{ℓ}	Mw		active power at the origin side of a power line	2	20	59
p_{ex}^{ℓ}	Mw		active power at the extremity side of a power line	2	20	59
v_{or}^{ℓ}	Kv		voltage at the origin side of a power line	2	20	59
v_{ex}^{ℓ}	Kv		voltage at the extremity side of a power line	2	20	59

12

Physical law compliance:

The predictions made by a surrogate model shall verify part or all of the following conditions and laws depending on the benchmark. Hence, the predicted values are shown by a hat above each symbol, e.g., $\hat{a}_{or}^{\ell}$ designates the prediction of current at the origin side of a power line l.

Table 2: Physical laws under several categories: basic ones regarding signs or common-sense values, uni-dimension ones when regarding only one kind of variable. Multi-dimension laws when considering several types will be added with future benchmark 3

ID	Type	Measure	Description	Bench #
		Basic		
P1	Current positivity	$\frac{1}{L}\sum_{\ell}^{L}\mathbb{1}_{(\hat{a}_{or,ex}^{\ell}<0.)}$	Proportion of negative current	1,2
P2	Voltage positivity	$\frac{1}{L}\sum_{\ell}^{L}\mathbb{1}_{(\hat{v}_{or,ex}^{\ell}<0.)}$	Proportion of negative voltages	2
P3	Losses positivity	$\frac{1}{L}\sum_{\ell}^{L}\mathbb{1}_{(\hat{p}_{ex}^{\ell}+\hat{p}_{or}^{\ell}<0.)}$	Proportion of negative energy losses	2
P4	Disconnected Line	$\frac{1}{L_{disc}}\sum_{\ell_{disc}}^{L_{disc}}\mathbb{1}_{(|\hat{x}_{ex}^{\ell}|+|\hat{x}_{or}^{\ell}|>0.)}$	Proportion of non-null a,p or q values	1,2
P5	Energy Losses	$\frac{\sum_{\ell=1}^{L}(\hat{p}_{ex}^{(\ell)}+\hat{p}_{or}^{(\ell)})}{Gen}\in[0.005,0.04]$	energy losses range consistency	2
		Uni-dimension law		
P6	Global Conservation	$\mathrm{MAPE}((Prod-Load)-(\sum_{\ell=1}^{L}(\hat{p}_{ex}^{\ell}+\hat{p}_{or}^{\ell})))$	Mean energy losses residual	2
P7	Local Conservation	$\mathrm{MAPE}((p_{k}^{prod}-p_{k}^{load})-(\sum_{l\in neig(k)}\hat{p}_{k}^{\ell}))$	Mean active power residual at nodes	2
P8	Voltage equality	$\sum_{\substack{i,j \\ i,j\in k \\ i\neq j}}\mathbb{1}_{(|v_i-v_j|>0)}$	Proportion of not equal voltages at nodes	2

Table 3: Thresholds table. The physical laws values are discretized into three intervals on the basis of theses thresholds. The values below the inferior threshold are considered as perfect. The values between the inferior and superior thresholds are considered as acceptable, and the values above the superior threshold are considered as not acceptable.

		Proportion based	Metric based		
Physics compliances		P1, P2, P3, P4, P8	P5	P6	P7
Thresholds	Inferior	95%	< 0.03 or > 0.05	5%	5%
	Superior	99%	$\in[0.005,0.04]$	10%	10%

Thresholds For physical laws, we define two thresholds, one to discriminate acceptable and non-acceptable law compliance, and the other one to discriminate acceptable and great compliance. For proportion-based metrics (P1,P2,P3,P4,P8) which returns a percentage of compliance, we set thresholds to 95% and 99% of compliance. For unit-based physical law metric, those thresholds are based on industrial relevance. For P4, energy losses on transmission grid are always above 0.5% losses and below 4%. So the compliance is deemed compliant if within this interval and not acceptable if away from those bounds by 25% (so if below 0.3% losses or above 5% losses). For conservation laws, we consider the average relative error (weighted mape) at two levels: at the observation level for global conservation (P6) and at the substation (node) level for local conservation (P7). We set two lower and upper bound thresholds to 5% and 10% respectively. The conservation is considered as great if lower than 5% and not acceptable if beyond 10%,

13

C.2 Pneumatic use case

We provide a description of the underlying physical model for the pneumatic use case, followed by a description of the physical aspects involved in each benchmark.

Physical problem The generic strong formulation of the problem is given by the following partial differential equations (PDEs)

Problem $\mathscr{P}$. *Find a displacement field $\boldsymbol{u}_\Omega : \Omega \times [0, T] \to \mathbb{R}^d$ and a constraint field $\boldsymbol{\Pi} : \Omega \times [0, T] \to \mathbb{M}^d$ such that*

$$\boldsymbol{\Pi} = \partial_{\mathbf{F}} W(\mathbf{F}) \qquad \text{in} \quad \Omega \times (0, T), \tag{1}$$

$$\text{Div}\,\boldsymbol{\Pi} + \boldsymbol{f}_0 = \mathbf{0} \qquad \text{in} \quad \Omega \times (0, T), \tag{2}$$

$$\boldsymbol{u}_\Omega = \boldsymbol{u}_{\Gamma_1 d} \qquad \text{on} \quad \Gamma_1 \times (0, T), \tag{3}$$

$$\boldsymbol{\Pi}\boldsymbol{v} = \boldsymbol{f}_2 \qquad \text{on} \quad \Gamma_2 \times (0, T), \tag{4}$$

$$\boldsymbol{u}_{\Omega v} \le g, \quad \Pi_v \le 0, \quad (u_{\Omega v} - g)\Pi_v = 0 \qquad \text{on} \quad \Gamma_3 \times (0, T), \tag{5}$$

$$\begin{cases} \|\boldsymbol{\Pi}_\tau\| \le \mu\,|\Pi_v|, \\ -\boldsymbol{\Pi}_\tau = \mu\,|\Pi_v|\,\dfrac{\boldsymbol{u}_{\Omega\tau}}{\|\boldsymbol{u}_{\Omega\tau}\|} \text{ if } \boldsymbol{u}_{\Omega\tau} \neq \mathbf{0}. \end{cases} \qquad \text{on} \quad \Gamma_3 \times (0, T). \tag{6}$$

- Equation (1) is the behaviour law, the relation between the stress acting on a body and the displacement
- Equation (2) is the equilibrium where $\boldsymbol{f}_0$ are the body forces acting on Ω (for instance gravity).
- $\boldsymbol{u}_{\Gamma_1 d}$ in equation (3) is the prescribed displacement on the Γ_1 boundary
- $\boldsymbol{f}_2$ in equation (4) are the normal traction forces acting on the Γ_2 boundary
- Equation (5) describes the unilateral contact conditions
  - Non-penetration, where g is the normal distance between the body and the foundation
  - Compressibility, the body is always compressed on the boundary
  - Compatibility, states that either there is contact ($\Pi_v < 0$ and $u_v = g$) or there is no contact ($\Pi_v = 0$ and $u_v < g$)

  These conditions are equivalent to assuming the foundation is perfectly rigid
- Equation (6) is the Coulomb's law of dry friction, with μ the friction coefficient.

Physical model differences between benchmarks

Benchmark 1

- Static problem
- Behaviour law: Linear elasticity

$$W = \frac{\lambda}{2} tr^2(\varepsilon)\mathbf{I} + \mu\,tr(\varepsilon^{\mathbf{2}})$$

where λ and μ are material parameters (Lamé coefficients).

Benchmark 2

- Quasi-static problem
- Behaviour law: Incompressible Mooney-Rivlin

$$W = c_1(j_1(\mathbf{C}) - 3) + c_2(j_2(\mathbf{C}) - 3)$$

where

$$j_1(\mathbf{C}) = i_1(\mathbf{C})\,i_3(\mathbf{C})^{-\frac{1}{3}} \qquad j_2(\mathbf{C}) = i_2(\mathbf{C})\,i_3(\mathbf{C})^{-\frac{1}{3}}$$

and c_1 and c_2 are material parameters.

- Rolling condition: it is enforced as a boundary condition applied at the rim, at each time step, and is described by the following equations:

$$\boldsymbol{u}_{\Gamma_1 d}(\boldsymbol{X}, t) = [V t; -d] + R(\omega t)\boldsymbol{X} - \boldsymbol{X} \quad \text{on } \Gamma_1 \times (0, T)$$

where

14

- V is the horizontal tire velocity with respect to the ground
- ω is the angular speed
- t is the instant of interest
- d is the vertical displacement
- R is the rotation matrix (2-dimension case)

$$R(\theta) = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \tag{7}$$

Input and output physical variables: We are now in position to provide similar tables and description for the pneumatic use case.

Table 4: Input and output variables of the model

Variable	Unit	Role	Description	Benchmark	Dimension
f_2	MPa	inputs	Force applied on the rim	1	1
t	s		Time instant	2	1
u_Ω	mm	outputs	Displacement field	1,2	4324,24936
λ_c	MPa		Contact stress field	2	864

Physical compliance: We provide similar tables and description for the pneumatic use case.

The predictions made by a surrogate model are not enough on their own to assess the accuracy of this very model. In order to investigate these aspects even further, the physical consistency of the predictions is evaluated against several physically-motivated criteria. Those criteria are also of interest in the practical application. Note that, as those criteria also rely on the underlying physical model, they are computed by the physical solver with the augmented simulator prediction, for simplicity.

Table 5: Physically-motivated criteria for the pneumatic use case

ID	Type	Measure	Description	Bench #
P1	Von-mises criteria	$\max_\Omega(\sqrt{\frac{3}{2}\sum_{1\le i,j\le 3}\sigma_{ij}^{\mathrm{dev}}\sigma_{ij}^{\mathrm{dev}}})$	Maximum of the Von-mises equivalent stress field over the domain	1,2
P2	Horizontal displacement	$\max_\Omega(u_\Omega\cdot\overrightarrow{x})$	Maximal displacement along the x-axis	1
P3	Vertical displacement	$\max_\Omega(u_\Omega\cdot\overrightarrow{y})$	Maximal displacement along the y-axis	1
P4	Tangential contact force	$\left(\int_{\Gamma_3}\lambda\,ds\right)\cdot\tau$	Resulting tangential force arising from the contact reaction	2
P5	Normal contact force	$\left(\int_{\Gamma_3}\lambda\,ds\right)\cdot v$	Resulting tangential force arising from the contact reaction	2

15

Table 6: Thresholds table. The physical laws values are discretized into three intervals on the basis of theses thresholds. The values below the inferior threshold are considered as perfect. The values between the inferior and superior thresholds are considered as acceptable, and the values above the superior threshold are considered as not acceptable.

		Proportion based				
	Physics compliances	P1	P2	P3	P4	P5
Thresholds	Inferior	80%	80%	90%	80%	90%
	Superior	95%	85%	95%	85%	95%

D Comparative related work table explained

In this section, we provide more details about the comparative table of related works (Table 7). We assess each prior work according to various considerations: 1) the set of evaluation criteria categories it covers (among ML-related, industrial readiness, OOD generalization, physics compliance); 2) efforts brought in terms of improved readability and real-world impact characterisation; 3) the environment setup including the considered reference physical simulator and dataset. We assign a grade of fully, partially, or not taken into account to each criterion.

From Table 7, while previous works have frequently covered each evaluation criteria category individually, none has actually covered the full range of evaluation criteria categories. LIPS unifies the evaluation setup in that regard, pushes it further in terms of Physics Compliance monitoring, and emphasizes the importance of OOD generalization and Industrial Readiness for ML researchers. Indeed, only one previous ML-related paper [7] has clearly considered OOD Generalization evaluation, and designed a specific test dataset with more complex grid topology changes not observed during training. It is only partially met in [8] as claims of generalization over unseen topologies are made but the corresponding evaluation setup and results are unclear. For non-ML papers, OOD Generalization is achieved by design as those models are tested on any kind of grids. Regarding industrial readiness, it is disregarded when no specific industrial application is actually considered for the simulation. In other cases, it is only partially met as they don't recalibrate the inference time according to the relevant application inference batch-size. Nonetheless in those cases, application-wise scalability or limited training data consideration exist. Finally regarding physics compliance, some papers implicitly evaluate it by design by training a physics-informed model or by simulating some physical equations in an optimization problem. However, none ultimately assess the performance in light of all pertinent physical laws for a specific application.

We stress that LIPS is the first to focus more on how a given level of performance will impact real applications. This is accomplished by applying appropriate thresholding to each evaluation criterion. These thresholds are calibrated with domain knowledge considerations for a given application as described in section C. The standardization of the visualization of evaluation across multiple non-homogeneous dimensions has not yet been addressed by any other work. It could be the case that the need for it was weaker, as often one or two criteria were the main focus of attention in simpler evaluation schemes.

Finally, regarding the environment setup, datasets represent more or less realistic distributions regarding grid injections or grid topologies. The dataset size is sometimes limited. Moreover, datasets are not always shared, but at least some data generation code and some documentation are sometimes provided. Only LIPS checks all those considerations, while others meet them partially. We raise a red flag when no access to the data distributions exists. In terms of reference simulator, the fastest one is not always the one used for reference (albeit they all have the same level of accuracy on IEEE grid test cases). We make sure to select the fastest open-source one for meaningful speed-up evaluation. About the models used, very few are genuinely available, maintained and kept for re-use and additional comparison. With LIPS, we ensure that baselines are shared and kept up-to-date. Last but not least, a benchmark is also stronger when diverse set of baselines are evaluated beyond one's model comparison with the reference simulator. Several papers present a *diverse* set of baselines mixing different ML-models, heuristics, reduced-ordered

16

Table 7: Comparative table between LIPS and related work for the power grid case. This highlights that our framework offers a comprehensive evaluation setup that was often barely covered by others.

	Reference	Evaluation criteria categories				Impact	Readability	Environment setup			
		ML-related	Industrial Readiness	OOD Generalization	Physics Compliances	Meaningful thresholding	Standard comparative visualization	Dataset	Ref physical Simulator	ML model repository	Baselines
ML-model papers	LeapNet [7] + [9]	Yes	Partial	Yes	No	No	No	Large, simple prod & varied topo distributions	Fast - Hades2 (proprietary)	Maintained	Yes (diverse)
	GNS/DSS [10, 11, 11]	Yes	No	No	Partial	No	No	Data generation, shared, simple distributions	Slow PandaPower (open source)	Not maintained	Yes (diverse)
	Fast Contingency analysis [12]	Yes	Partial	No	No	No	No	Not shared, realistic prod simple topo	Slow PandaPower (open source)	No access	Yes (diverse)
	Physics informed GNN [8]	Yes	No	Partial	Partial	No	No	Not shared, realistic prod simple topo	Slow PyPower (open source)	No access	Yes (uniform)
	Gridwarm [13]	Yes	Partial	No	Partial	No	No	Data generation, shared, simple distributions	None	Not maintained	Yes (diverse)
Benchmark suites	LIPS	Yes	Yes	Yes	Comprehensive	Yes	Yes	Large, doc, realistic prod topo distributions	Fast LightSim2Grid (open source)	Maintained	Yes (diverse)
	SimBench [14]	No (Optimization & heuristic)	Yes	Yes	No	No	No	Medium, doc, realistic prod simple topo	Slow PandaPower (open source)	No ML	Yes (diverse)
	PowerGridLib [15]	No (optimization)	Partial	Yes	Partial	No	No	Small, doc, realistic prod static topo	Med-speed PowerModel (open source)	No ML	Yes (uniform)

physical models or optimization. Some only present variations of a same kind of model, let say a given Deep Neural Network architecture, that we consider as a *uniform* set of baselines, which we tag as partial evaluation in that case.

E   Logistics & available materials

E.1   Datasets

The provided datasets are available through the `reference_data` folder. For each use case, a separate directory is considered. Concerning the power grid use case, we have considered two different grids, namely `L2RPN_case14_sandbox` (14 nodes) and `L2RPN_neurips_2020_track1_small` (36 nodes), through the experimentation and their related data are provided. For each benchmark, four datasets are provided, which are :

- `train`: the dataset used for training purpose of an augmented simulator;
- `val`: the dataset that may be used for the validation purpose;
- `test`: the dataset that present the same distribution of train dataset, which is used for evaluation purpose;
- `test_ood_topo`: the dataset that presents out-of-distribution samples which have never been seen during the training phase of an augmented simulator. As an example for power grid use case, it presents higher order topology actions in comparison to training test (two simultaneous line disconnections instead of only one disconnection considered in train).

E.2   Getting started Notebooks

There are several Jupyter Notebooks provided which guide the users to use the platform for generating the datasets, evaluation of baseline models and training augmented simulators. Some auxiliary notebooks are also provided to cover the experimentations explained in the paper and to fine-tune the hyper parameters. More details concerning these notebooks are provided in the following:

1. `DesignBenchmarks`: This notebook shows how the configuration files could be manipulated to create and introduce new scenarios;
2. `GenerateData`: This notebook shows how to generate the datasets using the LIPS platform and also to visualize the generated datasets using provided functions;

17

3. `EvaluateBaseline`: This notebook shows how one can evaluate a physics based method (DC approximation in power grid use case) or an already trained augmented simulator (for example a baseline fully connected architecture);

4. `TrainAnAugmentedSimulator`: This notebook shows how to train an augmented simulator from scratch (e.g., fully connected or leapNet) and to evaluate its performance;

5. `Complete_example`: This notebook presents a complete example of the benchmarking pipeline from loading datasets, training augmented simulators, analyzing their convergence, evaluating their performances with respect to various evaluation criteria categories;

6. `Benchmark1_Weight_sustaining_wheel`: This notebook provides information concerning the first benchmark of pneumatic use case;

7. `Benchmark2_Rolling_Wheel`: This notebook provides the information concerning the second benchmark of pneumatic use case;

8. `Auxiliary folder`: Include some supplementary notebooks. They allow for example to benchmark the bigger environment for power grids, fine-tune the hyper parameters of surrogate models, etc.

We work continuously to add more comprehensive notebooks to explain the different aspects of the framework. So, the list of notebooks will be evolved over time with respect to introduce and cover the new functionalities.

E.3 Models

We have provided two directories for the trained augmented simulators, where users could save the learned models.

- `trained_baselines`: include the baseline models which are used to provide the results shown in the paper. They can be used to reproduce the performance results;

- `trained_models`: Other trained models by the users could be saved in this directory.

These directories are also organized with respect to use cases and datasets and are accessed in relative notebooks to import the already trained augmented simulators.

E.4 Framework organisation

The developed package is available under `lips` directory which includes all the modules, classes and utility functions. Herein, we summarize breifly the list of modules, for greater details, we refer the readers to the documentation of the platform. The scheme of the LIPS package is also provided in Figure 1 where we can observe the three main modules which are Data, Benchmark and Evaluation.

- `augmented_simulators`: This module provides the base classes for augmented simulators which could be based on `Tensorflow` or `Pytorch`. On the basis of each of these libraries, we have also provided some baseline models which are available in their respective sub-modules. The base classes provide some important functions to be able to build, train and evaluate models;

- `benchmark`: This is the main module of the framework which allow to perform the complete pipeline of the evaluation. For each use case, a specific `Benchmark` class is provided allowing to generate datasets, to evaluate augmented simulators and more;

- `dataset`: This module is where the datasets are created and maintained. It offers some functions to save, load and generate some data. For each use case, a specific dataset class is considered, because of heterogeneous nature of their respective dataset;

- `evaluation`: This module is where the different categories of evaluation criteria are applied on the pair of observations and predictions to report the required KPIs;

- `metrics`: The set of metrics to compute the machine learning and physics compliances are offered in this module;

18

- `Physical_simulator`: this module provides the physical solvers which at basis for the generation of synthetic data. Each use case has its own solver;
- `plot`: some plotting utilities are provided to be able to visualize the data and the model performances.

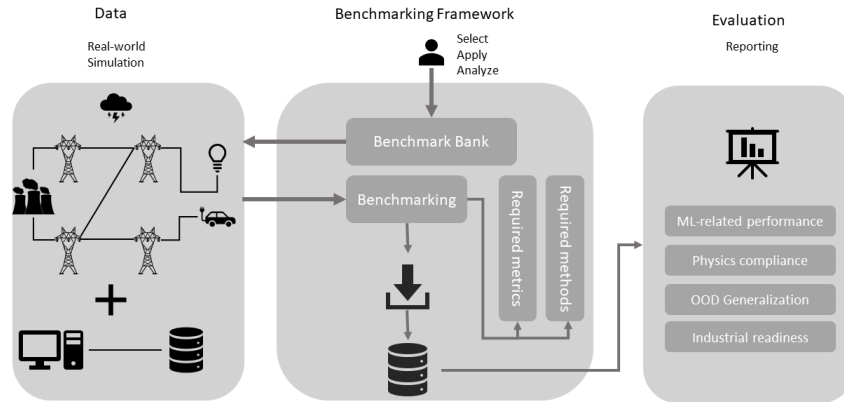



Figure 1: Benchmarking framework

E.5   Configuration Files

In order to give more flexibility to users to configure the benchmarks and augmented simulators, we have made use of configuration files. These configuration files are accessible from `configurations` directory of the repository root. Each use case has its own set of configuration files. We have also separated the configurations into two directories:

- **Benchmarks**: Configurations required to design a benchmark scenario
- **Simulators**: Configurations required to parameterize the hyper parameters of augmented simulators

In the following, we present firstly the configurations required to create a benchmark scenario. For this illustrative example, we show the configurations for power grid use case. Afterwards, we present an example of a configuration file for an augmented simulator. These configurations could be modified on-the-fly to change benchmark parameters or to propose a new augmented simulator with a set of completely new hyper parameters.

Benchmark configurations   As it can be seen below, each configuration file is composed of various sections. Each section could be identified by square brackets. For example, here we have shown the DEFAULT section with its set of options. It includes the generic default options for all the power grid related benchmarks. For example, we can use a specific grid environment that should be used for the experimentation, the data generation specific parameters which are indicated by the keys chronics, number of samples per chronic and also some seeds for reproducibility. We indicate also the variables (inputs and outpus), that should be stored during the generation of new data and also some very basic evaluation criteria that are considered for machine learning category (i.e., MSE and MAE).

19

```
[DEFAULT]
# The environment used for generation of data (IEEE14)
env_name = "l2rpn_case14_sandbox"
# Parameters used to initialize the environment
env_params = {
    "NO_OVERFLOW_DISCONNECTION": True,
    "MAX_LINE_STATUS_CHANGED": 999999,
    "MAX_SUB_CHANGED": 999999,
    "NB_TIMESTEP_COOLDOWN_LINE": 0,
    "NB_TIMESTEP_COOLDOWN_SUB": 0}
# Filter power grid chronics independently for each dataset
chronics = {
    "train": "^((?!(.*9[0-9][0-9].*)).)*$",
    "val": ".*9[0-4][0-9].*",
    "test": ".*9[5-9][0-4].*",
    "test_ood": ".*9[5-9][5-9].*"
    }
# Number of samples to be drawn from each chronic
samples_per_chronic = {
    "initial_chronics_id": 0,
    "train": 864,
    "val": 288,
    "test": 288,
    "test_ood": 288,
    }
# Seeds to be set for reproducibility
benchmark_seeds = {
    "train_env_seed": 1,
    "val_env_seed": 2,
    "test_env_seed": 3,
    "test_ood_topo_env_seed": 4,
    "train_actor_seed": 5,
    "val_actor_seed": 6,
    "test_actor_seed": 7,
    "test_ood_topo_actor_seed": 8,
    }
# Inputs and outputs to be used for training
attr_x = ("prod_p", "prod_v", "load_p", "load_q")
attr_tau = ("line_status", "topo_vect")
attr_y = ("a_or", "a_ex")
# Evaluation criteria per category
eval_dict = {
    "ML": ["MSE", "MAE"],
    "Physics": [],
    "IndRed": [],
    "OOD": []}
```

We show below also the configuration section (`Benchmark1`) with its set of options for the first benchmark of power grid use case. The power grid benchmark could be initialized and set using this configuration file. We indicate the set of attributes that should be stored when generating the data and also a set of parameters to design the specific scenario of this benchmark. The option `dataset_create_params` indicate a set of parameters, which are used by an agent (`XDepthAgent` available in dataset utils of the package) to make combinatory actions allowing to provide the required scenarios for each of the training, test and out-of-distribution datasets of this benchmark. These set of parameters are very specific to each benchmark. For example for this benchmark, we allow some topology action to change buses as the reference. For the training scenario, we authorize at maximum one line disconnection above of this reference topology. Test dataset will contain a line disconnection at each sample and finally the out-of-distribution will include 2 power line disconnections simultaneously which is not observed during the training phase.

We consider also a set of possible evaluation criteria for different categories. For example, for Machine learning related criteria, we consider MSE, MAE, MAPE and MAPE90 metrics and we would also compute the inference time with infinite batch size. For physics compliances we consider the current positivity verification. For industrial readiness, we measure also the inference time but from a real-world industrial point of view with a batch size 14000 which is indicated in evaluation metric parameters. Finally for generalization category, we consider the same set of criteria as machine learning.

20

```
[Benchmark1]
attr_x = ("prod_p", "prod_v", "load_p", "load_q")
attr_tau = ("line_status", "topo_vect")
attr_y = ("a_or", "a_ex")
dataset_create_params = {
    # REFERENCE PARAMS
    "reference_args" : {
        "topo_actions": [
                # topology change at substation 4
                {'set_bus':{'substations_id':[(4,(2,1,2,1,2))]}},
                # topology change at substation 1
                {'set_bus':{'substations_id':[(1,(1,2,1,2,2,2))]}},
                # topology change at substation 5
                {'set_bus':{'substations_id':[(5,(1,1,2,2,1,2,2))]}}
            ],
        # 70% unitary actions and 30% actions at depth 2
        "prob_depth": (0.7, 0.3),
        # only topology change actions
        "prob_type": (1., 0.),
        # include 20 percent DoNothing actions
        "prob_do_nothing": 0.2,
        # No line disconnections
        "max_disc": 0},
    "train": {
        # SCENARIO TOPOLOGY : disconnect or not one line at each tim step
        "prob_depth": (1.,),
        "prob_type": (0., 1.),
        "prob_do_nothing": 0.1,
        "max_disc": 1},
    "test":{
        # SCENARIO TOPOLOGY: disconnect one line at each time step
        "prob_depth": (1.,),
        "prob_type": (0., 1.),
        "prob_do_nothing": 0.,
        "max_disc": 1},
    "test_ood":{
        # SCENARIO TOPOLOGY: disconnect two lines at each time step
        "prob_depth": (0., 1.),
        "prob_type": (0., 1.),
        "prob_do_nothing": 0.,
        "max_disc": 2}
    }
# Evaluation criteria per category for Benchmark1
eval_dict = {
    "ML": ["MSE_avg", "MAE_avg", "mape_avg", "mape_90_avg", "TIME_INF"],
    "Physics": ["CURRENT_POS"],
    "IndRed": ["TIME_INF"],
    "OOD": ["MSE_avg", "MAE_avg", "mape_avg", "mape_90_avg", "TIME_INF"]}
# Set of adjustable parameters (tolerances) for evaluation criteria
eval_params = {
    # Inference batch size for industrial readiness evaluation
    "inf_batch_size": 14000,
    "EL_tolerance": 0.04,
    "GC_tolerance": 1e-3,
    "LC_tolerance": 1e-2,
    "KCL_tolerance": 1e-2,
    "ACTIVE_FLOW": True}
```

In the above configuration file, we parameterize an agent (called XDepthAgent) which allows to generate the required scenarios. Using this agent, we can perform some reference actions (indicated by reference_args) with its set of parameters and some scenario based actions, which are also vary with respect to the dataset (train, test, test_ood). The agent parameters are the following:

- prob_depth: the depth (number) and the corresponding probability. For example, the set (0.7, 0.3), indicates two actions possible at most, with higher probability corresponding to unitary actions (with probability 0.7);

- prob_type: indicates whether to perform topology (first item) or contingency action (second item) with a probability. For example, the set (1., 0.) indicates to perform only topology action (probability 1.);

21

- `prob_do_nothing`: A scalar value which indicates the proportion of do nothing actions to include. For example, a value 0.2 indicates that 20% of observations generated may contain no topology changes and no line disconnections;

- `max_disc`: the number of maximum power line disconnections authorized at each observation. For example, a number 2 indicates, at most 2 power lines could be disconnected at each observation.

Augmented simulator configurations   The augmented simulators which are used to model the physical phenomena could also be parameterized through configuration files. We show below an example of a configuration file for a Tensorflow fully connected model. In the repository, we provide also two other configuration files related to an implementation of Fully Connected architecture using Pytorch and the LeapNet using the Tensorflow library. As it can be seen, this configuration file could also have various sections and each section could be used independently to parameterize the required model. Through the `DEFAULT` configuration below, a number of hyper parameters of Fully Connected architecture could be adjusted: the number of layers and neurons, the activation function, drop-outs, metrics and loss function, batch size and number of epochs. Another variant of this default configuration is for example consider two layers with 100 neurons each that is suggested in section `CONFIG1`. This section could be set when training an augmented simulator.

```
# Configuration for a Fully Connected model implemented using Tensorflow library
[DEFAULT]
name = "tf_fc"
layers = (300, 300, 300, 300)
activation = "relu"
layer = "linear"
input_dropout = 0.0
dropout = 0.0
metrics = ["mae"]
loss = {"name": "mse",
        "params": {"size_average": None,
                   "reduce": None,
                   "reduction": 'mean'}}
device = "cpu"
optimizer = {"name": "adam",
             "params": {"lr": 3e-4}}
train_batch_size = 128
eval_batch_size = 128
epochs = 5
shuffle = True
save_freq = False
ckpt_freq = 50

[CONFIG1]
layers = (100, 100)
```

E.6   LIPS usage

Herein, the divers functionalities of LIPS through some working examples are described.

E.6.1   Generating data

One of the advantages of the proposed framework is to allow the generation of data. For this purpose, a physical solver for each use case is attached to the framework. For example, the Grid2op simulator is used for power grid use case and GetFem simulator is used for pneumatic use case. The generated states using these solvers are stored in datasets and are accessible for further required manipulations. The script below shows how to generate some data for the first benchmark of power grid use case. As it can be seen, the configuration file is used by `PowerGridBenchmark` class to initialize the set of parameters required for data generation. After instantiating the benchmark class, some samples are generated for each of required datasets (train, val, test and ood).

22

```python
# For power grid use case, import corresponding class
from lips.benchmark.powergridBenchmark import PowerGridBenchmark

# Indicate the paths to config file and a path where the data should be stored
CONFIG_PATH = PATH / TO / CONFIG / FILE / "l2rpn_case14_sandbox.ini"
DATA_PATH = PATH / TO / DATA / "l2rpn_case14_sandbox"
LOG_PATH = "lips_logs.log"

# Instantiate the benchmark class
benchmark1 = PowerGridBenchmark(benchmark_path=DATA_PATH,
                                benchmark_name="Benchmark1",
                                load_data_set=False,
                                config_path=CONFIG_PATH,
                                log_path=LOG_PATH)

# Generate some data for Benchmark1
benchmark1.generate(nb_sample_train=int(1e5),
                    nb_sample_val=int(1e4),
                    nb_sample_test=int(1e4),
                    nb_sample_test_ood_topo=int(1e4),
                    )
```

E.6.2 Training an augmented simulator

Once, we have generated some datasets, they could be used by augmented simulators for training. Hereafter, we show how to train a Tensorflow based augmented simulator versus a Pytorch based augmented simulator.

For Tensorflow based augmented simulators, you can simply import a required model and instantiate it using the provided configurations. Finally, using the `train` function, you can train the augmented simulator using the generated data for learning and validation. One should also pass a scaler as argument, to be able to preprocess the data.

```python
# Training a Tensorflow based augmented simulator
from lips.augmented_simulators.tensorflow_models import TfFullyConnected
from lips.dataset.scaler import StandardScaler

# indicate required paths
DATA_PATH = PATH / TO / DATA / "l2rpn_case14_sandbox"
BENCH_CONFIG_PATH = PATH / TO / CONFIG / "l2rpn_case14_sandbox.ini"
SIM_CONFIG_PATH = PATH / TO / SIM / CONFIG
LOG_PATH = "lips_logs.log"

# Instantiate a Tensorflow based augmented simulator
tf_fc = TfFullyConnected(name="tf_fc",
                         bench_config_path=BENCH_CONFIG_PATH,
                         bench_config_name="Benchmark1",
                         sim_config_path=SIM_CONFIG_PATH / "tf_fc.ini",
                         sim_config_name="DEFAULT",
                         scaler=StandardScaler,
                         log_path=LOG_PATH)

# Train it by indicating required datasets and some on-the-fly parameters
tf_fc.train(train_dataset=benchmark1.train_dataset,
            val_dataset=benchmark1.val_dataset,
            epochs=100
            )
```

For Torch based architectures, we provide a general purpose `TrochSimulator` class, which allows to train an augmented simulator and to predict using the trained model. However, the models should be derived from `torch.nn.Module` and passed an argument to this class. A GPU hardware could also be easily used if one is available. Otherwise, the CPU option should be selected.

23

```
# Training a pytorch based augmented simulator
from lips.augmented_simulators.torch_models.fully_connected import
    TorchFullyConnected
from lips.augmented_simulators.torch_simulator import TorchSimulator
from lips.dataset.scaler import StandardScaler

# indicate required paths
DATA_PATH = PATH / TO / DATA / "l2rpn_case14_sandbox"
BENCH_CONFIG_PATH = PATH / TO / CONFIG / "l2rpn_case14_sandbox.ini"
SIM_CONFIG_PATH = PATH / TO / SIM / CONFIG
LOG_PATH = "lips_logs.log"

# Instantiate a torch simulator (controller)
torch_sim = TorchSimulator(name="torch_fc",
                           # Indicate the model (class) that should be used
                           model=TorchFullyConnected,
                           # Indicate a Scaler class
                           scaler=StandardScaler,
                           log_path=LOG_PATH,
                           # use "cpu" if no GPU available
                           device="cuda:0",
                           seed=42,
                           bench_config_path=BENCH_CONFIG_PATH,
                           bench_config_name="Benchmark1",
                           sim_config_path=SIM_CONFIG_PATH / "torch_fc.ini",
                           sim_config_name="DEFAULT"
                           )
# train the TorchFullyConnected model by indicating required datasets
torch_sim.train(train_dataset=benchmark1.train_dataset,
                val_dataset=benchmark1.val_dataset,
                save_path=None,
                epochs=100,
                train_batch_size=128)
```

E.6.3 Evaluation using Benchmark class

Once the data is generated and an augmented simulator is trained, the benchmark class provide evaluation function (`evaluate_simulator`) allowing to compute all the required KPIs. This function takes as input the trained augmented simulator (here `torch_sim` object) trained in code snippet above, the dataset on which evaluation should be performed (here `all` means three datasets: valid, test, test_ood), some on-the-fly evaluation parameters (e.g., evaluation batch size) and if the results should be saved on disk by indicating corresponding paths. The results of evaluation is stored in a python dictionary for each evaluation metric categories.

```
# For power grid use case, import corresponding class
from lips.benchmark.powergridBenchmark import PowerGridBenchmark

# Indicate the paths to config file and a path where the data should be stored
CONFIG_PATH = PATH / TO / CONFIG / FILE / "l2rpn_case14_sandbox.ini"
DATA_PATH = PATH / TO / DATA / "l2rpn_case14_sandbox"
LOG_PATH = "lips_logs.log"

# Instantiate the benchmark class
benchmark1 = PowerGridBenchmark(benchmark_path=DATA_PATH,
                                benchmark_name="Benchmark1",
                                load_data_set=False,
                                config_path=CONFIG_PATH,
                                log_path=LOG_PATH)

# Evaluate the learned augmented simulator
torch_sim_metrics = benchmark1.evaluate_simulator(augmented_simulator=torch_sim,
                                                  eval_batch_size=128,
                                                  dataset="all",
                                                  shuffle=False,
                                                  save_path=None,
                                                  save_predictions=False
                                                  )

# print the performance metrics for test dataset
print(torch_sim_metrics["test"])
```

24

In order to ease the use of the platform for the users, some getting started Jupyter Notebooks are provided. These Notebooks explain in greater details, how one could generate some data from a desired power network, how to train an augmented simulator and how to evaluate the performance using the required KPIs.

F   Experimental settings

F.1   computing resources

All the experiments in the following sections are performed using a server equipped with AMD EPYC 7502P 32-Core Processor, NVIDIA RTX A6000 GPU and 128 GB of RAM. All computation time evaluation are run on the CPU with time measured per simulation or prediction. The augmented simulators use GPU for training, while power flow solvers and the inference phase of augmented simulators use CPU for computations. The experimentation results are reported using Tensorflow library (however, for the convenience, the users have the choice to use either Pytorch or Tensorflow libraries.

Training time   We have also investigated the training time required for both augmented simulators used in power grid use case. For Fully Connected augmented simulator, it takes around 45-60 minutes over 400 iterations (epochs) and for LeapNet, it takes around 50-70 minutes to be trained. The difference in training time between these approaches could be explained by the fact that LeapNet use a little more complicated architecture in comparison to Fully connected augmented simulator.

F.2   Augmented simulator visualization

We show the architecture used for LeapNet augmented simulator in Figure 2. It can be seen that the topology vector (labelled `tau`) intervenes in latent space using a Leap layer. This particular architecture shows nice generalization performances in comparison to fully connected architecture.

F.3   Preprocessing

No specific preprocessing is done in the raw datasets beside removing divergent simulation instances. However for training models, some functions are provided to run possibly useful preprocessing steps. current preprocessing step allows to encode the network topology in different ways. Standardization of injections and power flows are also available (zero mean and unit variance).

Various scalers are also provided in the dataset utilities which are:

- `StandardScaler`: A scaler which ensures to have a zero mean and unit variance;

- `PowerGridScaler`: A more advanced scaler, which is used for power grid used and LeapNet augmented simulator to encode the topology vector and other variable with respect to the power grid context.

These scalers could be used during the training phase of the augmented simulators by introducing them as an argument. They help to preprocess the data. The users could propose and use their own scaler as well and pass it as an argument to their own specific augmented simulator.

F.4   Hyper-parameter tuning

To set the hyper parameters, we have made use of tools like AutoKeras [16] and Nevergrad [17]. We are also investigating the integration of Optuna [18] as a tool for hyper parameter tuning to offer to the users of LIPS frameworks. As so, the researcher could focus only on important research topics (design of more competitive augmented simulators) rather than time-consuming operations as hyper parameter tuning.

25

26

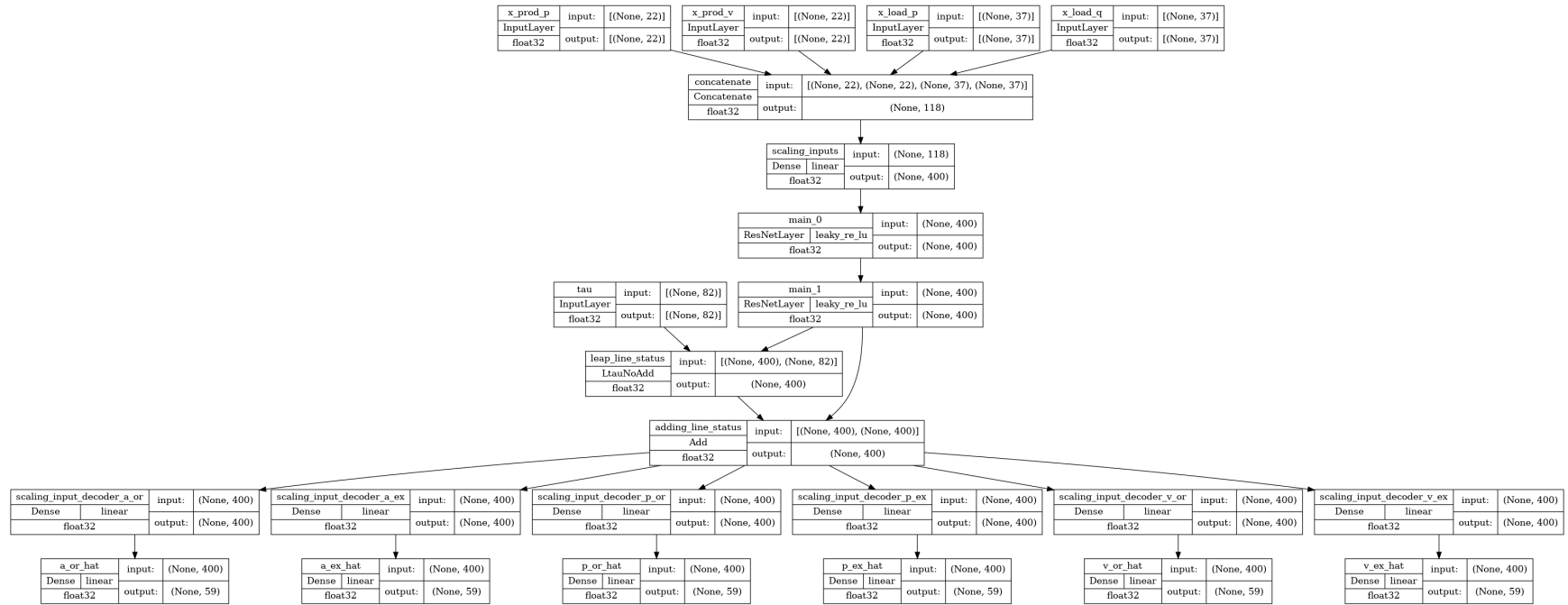



Figure 2: LeapNet augmented simulator architecture

G Complementary results

G.1 Numerical comparison table

Table 10 summarizes the evaluation results for both use cases over the different introduced scenarios. The values in this table are reported using the mean and standard deviation for each metric over 10 runs of the augmented simulators. Various evaluation criteria is used for machine learning category, which are: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Mean Absolute Percentage Error on 10% highest quantile values, and the speed up with respect to physical solver when the evaluation batch size is set to maximum possible value. Concerning the Readiness, we consider the Inference speed up using the batch sizes representing the real-world industrial use cases (much less than machine learning one). Finally, concerning the out-of-distribution category, the same set of metrics are computed using a dataset with samples presenting a different distribution than the training set for augmented simulators.

Quality Indicator threshold values for physical variables   In this work, in order to simplify the numerical results, we have categorized the performance of physics based solvers and augmented simulators on the basis of threshold values which are shown in Table 8. In the article, we have also associated a specific color to each category. For the variables with values below the inferior threshold, we consider perfect (green color) performance. For those with values between inferior and superior, we consider acceptable performance (yellow). Finally, we consider not acceptable (red color) for the values above the superior threshold.

Power grid   Thresholds for amperes A and active power P are given as a maximum percentage error. Operators takes margins of 5% when doing risk assessment (Benchmark 1) and are sensible to errors above that threshold. When searching for actions (Benchmark 2), they need more accuracy and this sensitivity thresholds is lower in the range of 1 and 2%. For voltages, we look at the absolute value error. Indeed voltages fluctuate around a nominal voltage value, and operators rather sees it in the form of a deviation that should be kept in a range of plus or minus 3kVolts. Hence 0.2kVolt error is fine but 0.5kVolt is not acceptable in that case.

Table 8: Thresholds table for variables. The values below the inferior thresholds are considered as perfect performance. The values between inferior and superior thresholds are considered as acceptable and the values above than the superior threshold are considered as not acceptable.

	Variables	Benchmark1 A	Benchmark2 A , P	Benchmark2 V
Thresholds	Inferior	5%	1.5%	0.2kV
	Superior	10%	5%	0.5kV

G.2 Physics compliances

The physics compliances are evaluated for benchmarks 1 and 2 respectively in Tables 9 and 12 for power grid use case. The DC approximation respect most of the physical laws, as it is based on physical laws by nature. It could be seen that more physical laws are computed for the second benchmark in comparison the first one. More variables are available for the second benchmark, hence, more complex physical laws should be verified consequently.

27

Table 10: Comparison table for benchmarks. The methods are Direct Current flow computation (DC), Fully Connected Neural Network (FC), LeapNet which is a specific neural architecture to encode the topology configurations and Unet, a convolutional neural network. For the Power Grid use case, variables are electrical current (A), Active power (P) and Voltage (V). For the pneumatic use case, variables are the displacement (u_Ω) and the contact stress (λ_c). The considered criteria are Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and its extended version which is computed on 10% of lines with highest current values (MAPE90), Inference time represented by the speed-up with respect to physic solver (Inf. speed-up), Out-of-distribution Generalisation (OOD Gen.), rate of violation of positive values (Pos.) and rate of incorrect predictions in the case of disconnected lines (Disc.)

| | | | | | Criteria category | | | | | | | |
| | | | | | ML-related | | | | Readiness | OOD Gen. | | |
Use case	Size	Benchmark	Methods	Variables	MAE	MAPE	MAPE90	Inf. speed-up	Inf. speed-up	MAE	MAPE	MAPE90
Power Grid	SMALL	Benchmark1	DC	A	-	-	.14	-	-	-	-	.15
			FC	A	-	-	2e-3 ± 3e-6	57	58	-	-	.14 ± 2e-5
			LeapNet	A	-	-	3e-3 ± 2e-5	11	12	-	-	.13 ± 1e-3
		Benchmark2	DC	A	-	-	.15	-	-	-	.16	-
				P	-	.09	-			-	.08	-
				V	-	.02	-			-	.02	-
			FC	A	-	-	3e-3 ± 1e-5			-	-	8e-2 ± 3e-4
				P	-	5e-3 ± 1e-5	-	62	-	-	.13 ± 7e-6	-
				V	6e-2 ± 8e-3	-	-			.43 ± 5e-2	-	-
			LeapNet	A	-	-	6e-3 ± 1e-4			-	-	.12 ± 2e-3
				P	-	9e-3 ± 4e-4	-	11	-	-	.28 ± 4e-3	-
				V	.18 ± 4e-2	-	-			.8 ± 5e-3	-	-
	BIG	Benchmark1	DC	A	-	-	.09	-	-	-	-	.09
			FC	A	-	-	5e-3 ± 4e-5	142	143	-	-	3e-2 ± 1e-4
			LeapNet	A	-	-	5e-3 ± 3e-5	28	28	-	-	3e-2 ± 3e-4
		Benchmark2	DC	A	-	-	.09	-	-	-	-	.10
				P	-	.08	-			-	.09	-
				V	-	.02	-			-	.03	-
			FC	A	-	-	1e-2 ± 4e-5			-	-	5e-2 ± 3e-4
				P	-	3e-2 ± 2e-4	-	130	-	-	.10 ± 3e-4	-
				V	.31 ± 9e-3	-	-			.50 ± 2e-2	-	-
			LeapNet	A	-	-	1e-2 ± 3e-5			-	-	5e-2 ± 3e-5
				P	-	2e-2 ± 3e-3	-	25	-	-	9e-2 ± 1e-5	-
				V	.40 ± 2e-3	-	-			.60 ± 1e-2	-	-
Pneumatic		Benchmark1	FC	u_Ω	2e-3 ± 3e-4	-	-	18	-	-	-	-
			UNet	u_Ω	1e-1 ± 8e-4	-	-	18	-	-	-	-
		Benchmark2	FC	u_Ω	23	-	-			23	-	-
				λ_c	2e-2 ± 2e-4	-	-	11	-	2e-2 ± 2e-4	-	-

28

Table 11: Color blind version of Table 3 of the article. Benchmark result table for the two use cases under 4 categories of evaluation criteria. The performances are reported using three colors computed on the basis of two thresholds. Colors and symbol meaning: ○ Not acceptable ● Acceptable ● Great ◎ two problem scales reported (in that case, speed-up for smaller scale is in parenthesis). The number of circles corresponds to the number of variables or laws that are evaluated. For quantitative values from which this table is derived, please refer to section G.1 of appendix.

				Criteria category				
				ML-related		**Readiness**	**OOD Gen.**	**Physics**
			Methods	Quality	Speed-up	Speed-up	Quality	Domain laws
Use cases	Power Grid	Bench1	DC	a ◎	NA	19 (7)	a ◎	P1 ◎
			FC	◎	19 (22)	17 (20)	◎	◎
			LeapNet	◎	17 (19)	14 (17)	◎	◎
		Bench2	DC	a p v ◎◎◎	NA	5 (3)	a p v ◎◎◎	P1 P2 P3 P4 P5 P6 P7 P8 ◎◎◎◎◎◎◎◎
			FC	◎◎◎	99 (157)	57 (27)	◎◎◎	◎◎◎◎◎◎◎◎
			LeapNet	◎◎◎	90 (140)	54 (24)	◎◎◎	◎◎◎◎◎◎◎◎
	Pneumatic	Bench1	FC	u_Ω ●	18	NA	NA	P1 P2 P3 ●●○
			UNet	○	18	NA	NA	○○○
		Bench2	FC	u_Ω λ_c ○●	11	NA	u_Ω λ_c ○●	P1 P2 P3 ○●●

29

Table 9: Physics compliance Table `BENCHMARK1`. The only physical metric considered for this benchmark is the proportion of violation of electrical current positive values

Grid	Evaluation	Methods	CurrentPos
SMALL	Test	DC approximation	0
		Fully Connected	2e-2 ± 2e-4
		LEAP Nets	2e-2 ± 7e-4
	OOD	DC approximation	0
		Fully Connected	2e-2 ± 2e-4
		LEAP Nets	3e-2 ± 7e-5
BIG	Test	DC approximatio	0
		Fully Connected	8e-3 ± 2e-4
		LEAP Nets	8e-3 ± 3e-4
	OOD	DC approximation	0
		Fully Connected	1e-2 ± 2e-4
		LEAP Nets	1e-2 ± 3e-5

Table 12: Physics Compliances Table `BENCHMARK2`. DC stands for Direct Current flow computation, FC is Fully Connected Neural Network and LeapNet is a specific neural architecture to encode the topology configurations. The Physics Compliance metrics are the rate of violation of positive values (Pos.) and the rate of incorrect predictions in the case of disconnected lines (Disc.)

Grid	Evaluation	Methods	Physics							
			CurrentPos	VolPos	LossPos	DiscLines	LossRes	GC	LC	VolEq
SMALL	Test	DC	0.	0.	0.	0.	0.	1.	.02	.53
		FC	1e-2	1e-2	.13	-	.23	1e-2	2e-3	.97
		LeapNet	1e-2	8e-3	.13	-	.20	1e-2	2e-3	.98
	OOD	DC	0.	0.	0.	0.	0.	1.	.02	.5
		FC	1e-2	1e-2	.16	-	1.37	6e-2	1e-2	.98
		LeapNet	1e-2	1e-2	.14	-	1.10	3e-2	1e-2	.99
BIG	Test	DC	0.	0.	0.	0.	0.	1.	.01	.02
		FC	1e-2	1e-2	.24	-	2.89	6e-2	8e-3	.99
		LeapNet	1e-2	1e-2	.20	-	1.70	2e-2	9e-2	.99
	OOD	DC	0.	0.	0.	0.	0.	1.	.02	.04
		FC	1e-2	9e-3	.25	-	9.08	8e-2	1e-2	.99
		LeapNet	1e-2	7e-3	.20	-	8.5	8e-2	1e-2	.99

We present the results for both benchmarks in the pneumatic case in table 13 and 14. It turns out that, from the physical consistency point of view, the augmented models don't behaves well at all. Such a thing can be explained by the size of the dataset considered and required more investigations in the near futures. However, one expected result is the accuracy of prediction for the contact stress in benchmark 2. Despite being a pure out-of-distribution case, the ML-model behaves surprisingly well. Unfortunately, the results are far from satisfactory in the same benchmark for the displacement; in our opinion, it means the scaler proposed to handle the out-of-distribution displacement prediction is not working as expected as the optimizer associated to the augmented simulator fails to converge properly.

Table 13: Physics Compliances Table `BENCHMARK1` pneumatic use case.

Pneumatic (%)	Evaluation	Methods	Physics		
			Von-Mises	HorizontalDisp	VerticalDisp
	Test	FC	20 ± 9	16 ± 3	46 ± 200
		Unet	318 ± 15	276 ± 52	381 ± 368

Table 14: Physics Compliances Table `BENCHMARK2` pneumatic use case.

Pneumatic (%)	Evaluation	Methods	Physics		
			Von-Mises	NormalContact	TangentialContact
	Test	FC	6e8 ± 7e8	7 ± 4	2 ± 1

30

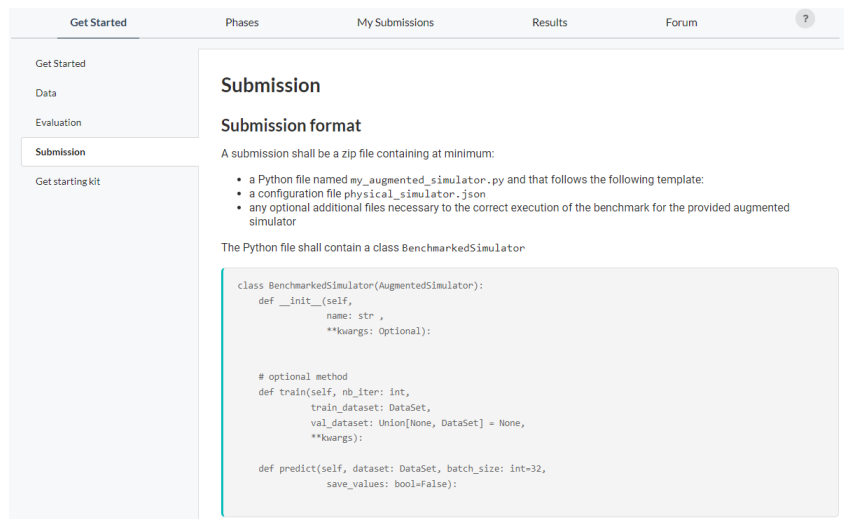
H Accessibility

H.1 Code repository and data availability

The data and developed platform "Learning Industrial physical simulation benchmark suite: the power grid case" can be accessed via the following public github repository : `https://github.com/IRT-SystemX/LIPS`.

H.2 Submission on Codabench

In order to ease the testing, reproducibility and evaluation of Augmented Simulators, we provide an instance of the LIPS framework running on the Codabench[19] platform; it can be accessed using the following link: `http://htmlpreview.github.io/?https://github.com/IRT-SystemX/LIPS/blob/main/codabench/codabench.html`.

We strongly encourage researchers and practitioners to use this service to assess the quality of their models against our diverse benchmarks.

The submission on Codabench is relatively simple, participants have to follow the following steps:

1. *Prepare the Augmented Simulator bundle* – This phase consists in implementing the augmented simulator to be tested. The implementation is quite free (meaning that the implementation of the augmented simulator is not restricted to a specific ML framework such as TensorFlow or PyTorch). However it has to comply with a "standard" interface so that it can be automatically instantiated and run by Codabench when it is submitted. The details of the interface are provided on LIPS Codabench page, section "Submission" (see Figure 3).

   A submission is a zip file shall contain at minimum:

   - a Python file named `my_augmented_simulator.py` and that contains a class named `BenchmarkedSimulator` that extends the LIPS class `AugmentedSimulator`:
   - a configuration file `simulator_config.json` that contains in particular specific parameters required during the initialization of the submitted augmented simulator
   - any optional additional files necessary to the correct execution of the benchmark with the provided augmented simulator.
   - (optionally) pre-trained simulator data.

   A submitted simulator is preferably pretrained; otherwise it may be (re)-trained on the Codabench platform depending on the availability of computing resources.

Figure 3: Preparation of an Augmented Simulator submission to Codabench

31

2. *Upload of submission on Codabench* – Once the augmented simulator bundle is ready, it may be submitted on Codabench (tab "My submissions"). The participant has to select the list of benchmark tasks he wants to test his simulator against (see Figure 4)

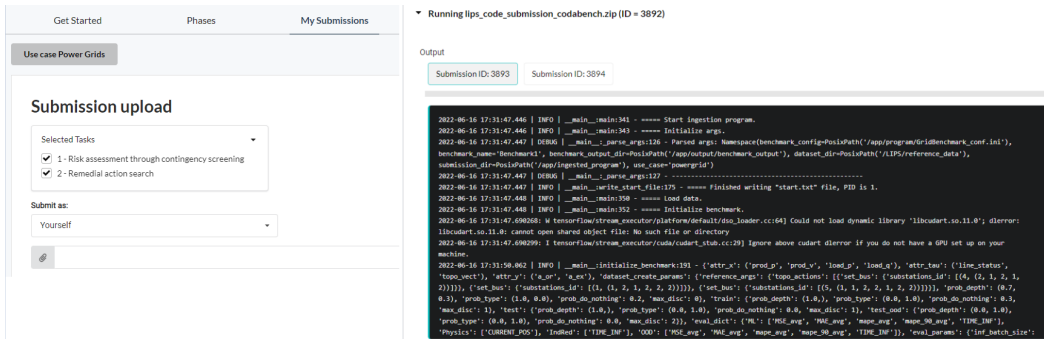



Figure 4: Submission and monitoring of an Augmented Simulator submission to Codabench

Based on this configuration, the selected benchmarks are instantiated, each on an isolated Docker container, and an instance of the provided augmented simulator is created for each selected benchmark. During the execution of the benchmark, a set of logs are displayed in order to check if it is running smoothly.

Once the execution is finished, the status of the submission is updated (see Figure 5: "Finished" in case of successful execution). It may be a "Failed" in some cases, for example if the provided Augmented Simulator implementation does not comply with the required interface or if a timeout occurs.

Figure 5: Status of an Augmented Simulator submission to Codabench

3. *Obtain results and ranking* – A leaderboard is then available (Figure 6), on which the user can see the results of his Augmented Simulator for the selected benchmarks and according to the different criteria we explained in this paper. This table allows to compare with other submissions.

Figure 6: Leaderboard for Power System Benchmark 1 on Codabench

Potential issues or bugs found when using the LIPS Codabench instance may be reported in the LIPS github as issues: `https://github.com/IRT-SystemX/LIPS/issues`.

32

I Platform documentation

The platform's documentation is available at `https://lips.readthedocs.io/en/latest/`. All the classes, variables and corresponding functions are explained in greater details with more examples.

J Computation time investigation

To investigate the computation time of physical solvers and augmented simulators, we consider the whole inference procedure which consists in the following three steps:

- Pre-processing: the time required for the data to be prepared as to respect the right format (dimension) and scale used by a physical solver or an augmented simulator;
- Solving/Inference: the time required by a physical solver or an augmented simulator to solve one instance of the problem (one powerflow for power grid use case);
- Post-processing: The time required to reformat or re-scale the results.

In the following section, we provide more details concerning how the computation time is computed for physical solvers and augmented simulators.

J.1 Physical solvers

The time spent in the solver can be split into different "steps" that depend on the actual implementation of the solver.

1. This can include some pre-processing step(s)
   - transfer between RAM and GPU (if computing happens on GPU)
   - data copied in the RAM:
     - data are sent to the python virtual machine by grid2op and could need to be transferred to the JVM if solver is written in java for example
     - data can be copied somehow if the solver if not "fully integrated" to grid2op
   - transfer between RAM and "network card": if the solver is accessed through a REST API for example
   - serialization / deserialization of the send by grid2op to be "understood" by the solver
   - computation of some initial data
   - etc.

   This steps depends on the technology used by the solver and also the implementation of the "Backend" class (it can be more or less optimized) as well as the internal data representation used by the solver (which may be different from grid2op and require more or less "copy" and serialization / deserialization) etc.

2. Then the computation of the powerflow takes place. This depends "only" on the solver used.

3. Then, finally, post-processing step(s) can happen:
   - conversion between some internal variables and output variable (eg computation of the flows knowing all the complex voltages at each nodes)
   - transfer between GPU and RAM
   - data copied in the RAM
   - transfer from "network card" to RAM (eg. if using a backend connected to a REST API)
   - serialization / deserialization of the data to be consistent with grid2op representation
   - etc.

   As for step 1 above, this depends only on the actual implementation of the backend (if read from grid2op). The most important part for this "post processing step" would be the possibility to get the results (mainly the flows) in a convenient format (numpy / pandas / etc.) in python.

33

Solver time And of course it is still possible to "go down a level" by splitting the "compute a powerflow" into different steps that, this time, heavily depends on the implementation of the solver.

For example, for lightsim2grid:

- α the topology / shunts of the grid is transformed into a Ybus matrix (the admittance matrix)
- β the productions / loads are transformed into a Sbus vector
- β (bis) Get initial guess for internal (solver) variables (eg V, theta)
- γ Ybus, Sbus and the initial guess are then given to a "Powerflow solver" that computes a powerflow and output (V, theta) at each bus (each row / column of Ybus)
- $delta$ (V, theta) at each bus are then transformed to flows, voltages, production values etc. and can then be read by grid2op

Security analysis For example, in real time, each 5 minutes a "security analysis" is run. This consists in computing a first time (with a full grid) α, β, γ, δ in order to get a good initial state (and reusing the matrix / vector computed in α and β). Then you slightly modify the matrix obtained after α and recompute γ and δ. You can do this last part "asynchronously" ("in parallel") as each computation is independent. This is the case for the first benchmark of power grid use case.

J.2 Augmented simulators

To evaluate the computation time of augmented simulators, we compute the time passed by the methods during the `predict` function of augmented simulators. This function shall include the three above-mentioned steps. An example of this function for `Tensorflow` based models is shown below.

```python
def predict(self, dataset: DataSet, **kwargs) -> dict:
    """_summary_

    Parameters
    ----------
    dataset : DataSet
        test datasets to evaluate
    """
    # preprocess the data
    processed_x, _ = self.process_dataset(dataset, training=False)

    # make the predictions
    predictions = self._model.predict(processed_x, batch_size)

    # post process the data
    predictions = self._post_process(dataset, predictions)

    return predictions
```

K Batch Size considerations for industrial Applications

As we can see in Figure 7, the speed-up factor depends on the batch size and increases with the batch size. Indeed for small batch sizes, there exists some overhead time unrelated to the pure inference time that needs to be considered. This overhead time becomes negligible as the batch size increases. In industrial applications, only a given amount of simulations might be run simultaneously. This hence define the target batch size for this application. And the speed-up factor needs to be evaluated in that context to be relevant for that application.

Benchmark 1 For the power grid risk assessment, operators perform near real-time at a given time simulations of security analysis (simulating every line contingency) over a temporal horizon of about 100 timesteps (equivalent to a day horizon) and for difference reference topologies. Hence the Batch size is of the order of:

$Batch\_size = n_{lines} * 100 * n_{topo\_ref}$

34

Benchmark 2   For remedial action search, operators are looking for solutions on a given situation when there is a risky contingency that creates an overload. There are about 7-10% of such risky contingencies for a given situation. There is usually about 20 to 50 known remedial action candidates for a given risky contingency. Hence the Batch size is of the order of:

$$Batch\_size = n_{lines} * 10\% * n_{actions}$$

Figure 7: Inference Batch-size time complexity using LeapNet

L   Further investigations (beside the provided developments)

L.1   Convergence

Hereafter, we have presented the convergence behavior of the augmented simulators (see Figure 8). In these figures, we have presented the loss curves related to the training (blue curve) and to the validation steps (orange curve) of augmented simulators computed over 200 epochs. As it can be observed in these figures, the both models (FC and LeapNets) have been converged over 200 epochs. The validation curve remains relatively close to the training curve and we do not observe an abrupt increase in its behavior, which confirms the fact that the models have not overfitted during the training phase. The difference between two models resides in the fact that the LeapNet model starts with higher losses than the Fully Connected model at the beginning, and it gradually reduces over time.

(a) Fully Connected model   (b) LeapNet model

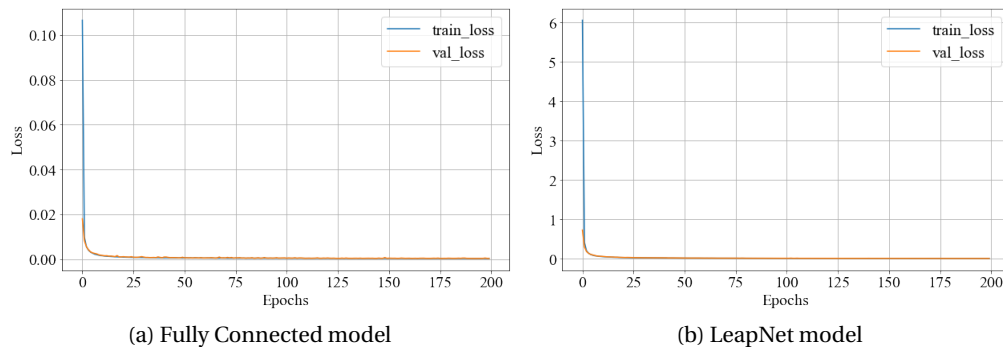



Figure 8: Analysis of the convergence behavior of the augmented simulators

35

L.2 Investigation of required data volume for online learning

In the real-world, it is also in interest to determine the required volume of data for obtaining a desired inference precision. To perform this benchmark, the data volume is varied between 1,024 and one million samples and the averaged NRMSE measure is reported for each data volume (see Figure 9). It should be noted that the models were configured to learn progressively (learning on-the-fly) as the samples are observed over time. It is a requirement for power network industries to be able to train the models in this way, as the space of possible topological actions is humonguous and all the possible situations could not be included in the training set. From these figures, it can be observed that all the experimented models require at least about 400,000 data samples to converge.

(a) Fully connected	(b) ResNet	(c) LeapNet

Figure 9: Data volume benchmark

References

[1] Antoine Marot, Benjamin Donnot, Camilo Romero, Balthazar Donon, Marvin Lerousseau, Luca Veyrin-Forrer, and Isabelle Guyon. Learning to run a power network challenge for training topology controllers. *Electric Power Systems Research*, 189:106635, 2020.

[2] Antoine Marot, Benjamin Donnot, Gabriel Dulac-Arnold, Adrian Kelly, Aïdan O'Sullivan, Jan Viebahn, Mariette Awad, Isabelle Guyon, Patrick Panciatici, and Camilo Romero. Learning to run a power network challenge: a retrospective analysis. *arXiv preprint arXiv:2103.03104*, 2021.

[3] Grid2op. https://github.com/rte-france/Grid2Op.

[4] Lightsim2grid. https://github.com/BDonnot/lightsim2grid.

[5] Julien Gillard. *An Efficient Partitioned Coupling Scheme for Tire Hydroplaning Analysis*. 05 2018.

[6] Yves Renard and Konstantinos Poulios. Getfem: Automated fe modeling of multiphysics problems based on a generic weak form language. *ACM Transactions on Mathematical Software*, 47:1–31, 12 2020.

[7] Balthazar Donon, Benjamin Donnot, Isabelle Guyon, Zhengying Liu, Antoine Marot, Patrick Panciatici, and Marc Schoenauer. Leap nets for system identification and application to power systems. *Neurocomputing*, 416:316–327, 2020.

[8] Luis Böttcher, Hinrikus Wolf, Bastian Jung, Philipp Lutat, Marc Trageser, Oliver Pohl, Andreas Ulbig, and Martin Grohe. Solving ac power flow with graph neural networks under realistic constraints. *arXiv preprint arXiv:2204.07000*, 2022.

[9] Benjamin Donnot, Isabelle Guyon, Marc Schoenauer, Antoine Marot, and Patrick Panciatici. Anticipating contingengies in power grids using fast neural net screening. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[10] Balthazar Donon, Zhengying Liu, Wenzhuo Liu, Isabelle Guyon, Antoine Marot, and Marc Schoenauer. Deep statistical solvers. In *NeurIPS*, 2020.

36

[11] Balthazar Donon, Rémy Clément, Benjamin Donnot, Antoine Marot, Isabelle Guyon, and Marc Schoenauer. Neural networks for power flow: Graph neural solver. *Electric Power Systems Research*, 189:106547, 2020.

[12] Florian Schäfer, Jan-Hendrik Menke, and Martin Braun. Evaluating machine learning models for the fast identification of contingency cases. *Applied AI Letters*, 1(2):e19, 2020.

[13] Shimiao Li, Amritanshu Pandey, and Larry Pileggi. Gridwarm: Towards practical physics-informed ml design and evaluation for power grid. *arXiv preprint arXiv:2205.03673*, 2022.

[14] Steffen Meinecke, Džanan Sarajlić, Simon Drauz, Annika Klettke, Lars-Peter Lauven, Christian Rehtanz, Albert Moser, and Martin Braun. Simbench—a benchmark dataset of electric power systems to compare innovative solutions based on power flow analysis. *Energies*, 13:3290, 06 2020.

[15] Sogol Babaeinejadsarookolaee et al. The power grid library for benchmarking ac optimal power flow algorithms, 2021.

[16] Haifeng Jin, Qingquan Song, and Xia Hu. Auto-keras: An efficient neural architecture search system. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1946–1956, 2019.

[17] Jeremy Rapin and Olivier Teytaud. Nevergrad-a gradient-free optimization platform, 2018.

[18] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019.

[19] Zhen Xu, Huan Zhao, Wei-Wei Tu, Magali Richard, Sergio Escalera, and Isabelle Guyon. Codabench: Flexible, easy-to-use and reproducible benchmarking for everyone. *arXiv preprint arXiv:2110.05802*, 2021.

37