Automated Processing of Pipelines Managing Now- and Forecasting of Infectious Diseases

Shahbaz Memon*, Johann F. Jadebeck†¶, Michael Osthege†, Anna Wendler‡, David Kerkmann§, Henrik Zunker‡, Wolfgang Wiechert†¶, Katharina Nöh†, Jens Henrik Göbbert*, Björn Hagemeier*, Morris Riedel*||, Martin J. Kühn‡**

*Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Jülich, Germany, m.memon@fz-juelich.de

†Institute for Bio- and Geosciences, IBG-1: Biotechnology, Forschungszentrum Jülich GmbH, Jülich, Germany

¶Computational Systems Biotechnology (AVT.CSB), RWTH Aachen University, Aachen, Germany

‡Department for High-Performance Computing, Institute for Software Technology, German Aerospace Center, Cologne, Germany

§Helmholtz Centre for Infection Research, Brunswick, Germany

||School of Engineering and Natural Sciences, University of Iceland, Reykjavik, Iceland

**Life and Medical Sciences Institute, University of Bonn, Bonn, Germany

***Abstract*—When faced with the challenge of now- and forecasting infectious diseases, multiple data sources and state-of-the-art models have to be considered. Automatic aggregation, processing, and publishing to relevant data sinks is paramount to achieving consistent, reproducible, and timely results given daily-reported data. To facilitate scientific collaboration and reproducibility of workflows, open and extensible architectures for compute pipelines are required.**

In this research, we devise an architecture realizing the seamless management and processing of reproducible pipelines. Our case-study is a daily pipeline for nowcasting the state of SARS-CoV-2 in Germany based on public data and state-of-the-art models implemented in the simulation software MEmilio. The results of our pipeline are pushed to ESID (Epidemiological Scenarios for Infectious Diseases), a user interface to epidemiological simulations.

To realize the given pipeline, a workflow management system is required to ensure pipeline processing and secure access to multiple heterogeneous data storages. For this purpose, we based our work on an open-source workflow management system - Apache Airflow, which provides the orchestration, coordination and management of complex connected tasks. S3 is utilized as an intermediate data storage service for sharing data between workflow steps and persisting experiment output. We provide a comprehensive view on our work on automated, end-to-end and reproducible pipelines, with detailed commentary on use case, and its realization.

I. Introduction

As demonstrated by the recent COVID-19 pandemic, infectious diseases carry substantial risk of disrupting societies on a global scale [1], [2]. Non-pharmaceutical interventions (NPIs) are a key strategy to complement pharmaceutical or medical treatment and to protect populations. However, NPIs impact the daily lives of large groups of people and may temporarily curtail personal rights. Thus, NPIs need to be minimized for economic and social impacts [3]–[5]. Aside from retrospective analyses, as a consequence, interventions and their impact on infectious disease dynamics must be monitored for efficacy in regularly updated nowcasts. Furthermore, for timely reaction, continuous updates in response to dynamic situations need forecasting.

Two ingredients are required for a reliable now- and forecasting recipe. Firstly, accurate and robust statistical and mathematical models are required to describe the spread of the infectious disease as well as its ramifications on public health. Secondly, the practical application of these models requires complex, multistep workflows, and a robust compute and data infrastructure, including appropriate tools and technologies centered around a suitable workflow management system.

A broad body of mathematical-epidemiological modeling literature exists, which has only been expanded in the wake of the SARS-CoV-2 pandemic, see, e.g. [6]–[10]; in particular for Germany many different groups have used a variety of different models [11]–[19]. For the purpose of this paper, we consider the problem of accurately modeling the spread of infectious diseases to be solved sufficiently well.

The blueprint of processing these models comprises several steps, which may consist of data-staging, pre-processing, model-execution, and finally post-processing [20] of the results. In the data-staging step, data, such as daily case numbers, census, vaccination, and hospitalization data, is aggregated from various sources. Then, in compute-intensive tasks, the model generation goes through multiple processing steps of the data staged in the previous phase to compute the model output. Finally, the produced results are persisted and can be utilized further for data visualization purposes.

Typically, workflows conducting the simulation of infectious diseases are invoked either automatically, scheduled or based on conditional parameters. Some scenarios are, for instance, ingestion of new data or explicit re-run with variation of parameters reaching a certain threshold. Therefore, a workflow management system should be capable of supporting this requirement of managing and controlling

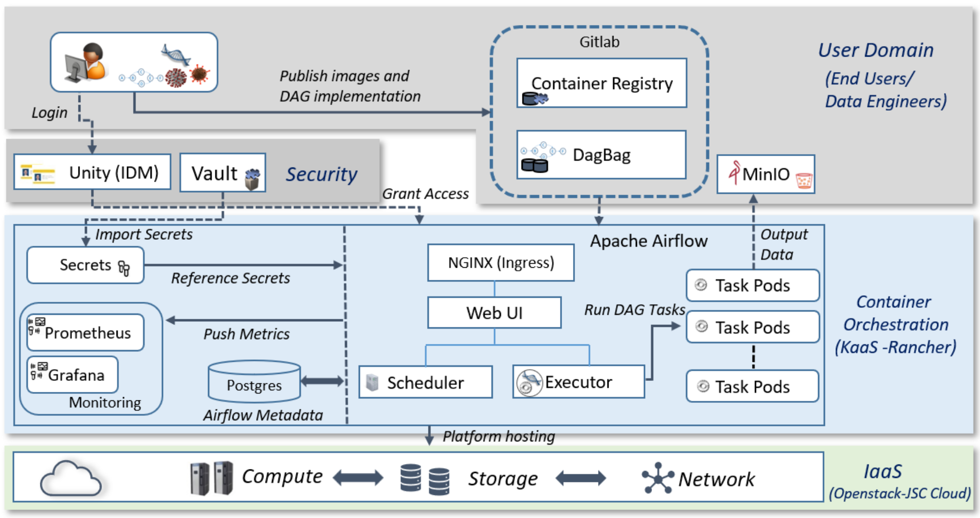

Fig. 1. Architecture showing the interaction of end user and Airflow deployment

pipeline execution with more control temporally, and also able to transfer data from or to different sources.

The envisioned pipelines are motivated by the FAIR and FAIR4RS principles, which are practical guidelines aimed at maximizing the value and quality of research data and software, respectively [21], [22] Generally, the usefulness of the FAIR and FAIR4RS principles is evident when considering sharing and reproduction of research data and software. The FAIR principles suggest that all used and produced data should be made available publicly to create trust and acceptance and to foster wider scientific cooperation. In the context of non-pharmaceutical interventions affecting the public, open data also helps justifying and explaining political decisions to the public. Regarding the FAIR4RS principles [21], for example, modular and reusable workflows support quick adaption of existing pipelines to new and unseen circumstances, such as new virus variants or the roll-out of vaccines. That said, the envisioned infrastructure and the pipeline should also be open and documented in a way that other scientific communities can reproduce and adapt the implementation in an interoperable and reusable manner.

Guided by these principles, we provide a multi-tier architecture using state-of-the-art and open-source infrastructure and workflow management system. To validate and demonstrate our architecture, we construct a daily and spatially resolved nowcasting pipeline for the state of COVID-19 in Germany based on publicly available data. The highlighted models are considered state-of-the-art and to be implemented in the software MEmilio [23] and visualized via the web application ESID [24] (Epidemiological Scenarios for Infectious Diseases). The initiative is part of the project LOKI [1](Local Control System for Infection Outbreaks). The focus of the LOKI project can be divided into four different areas. Firstly, as model outputs can only be as good as the data input, large efforts go into the collection and processing of data sets. Secondly, to create reliable model outputs, spatially and demographically resolved models integrating important effects like vaccination or waning immunity are developed. In order to have timely model outputs, all models are implemented in highly efficient C++, parallelized where possible, and, additionally, surrogate models based on artificial intelligence are developed. HPC infrastructure and visualization are the essence of the third work package of LOKI and finally, regular feedback loops and user experience studies are conducted with pilot local health authorities. LOKI aims to increase the pandemic preparedness in Germany, by giving local health authorities the tools to effectively monitor local outbreaks and simulate NPIs to see their effect in infectious disease dynamics' mitigation. Naturally, a workflow management system is required to serve the realization of aforementioned LOKI scenarios.

The rest of the paper is structured as follows. Section II gives a background on technologies involved. Section III discusses the related work. The architectural realization is described in Section IV, and Section V presents our scientific use case. We conclude the paper in Section VI.

II. Related Work

The objective of predicting infectious disease dynamics and to assess the effect of NPIs has seen largely increased with the emergence of Sars-CoV-2 [25]–[28]. Aside from LOKI, several research consortia with different foci in infectious disease modeling have been formed and brought together under the roof of the Modeling Network for Severe Infectious Diseases (MONID) [2]. Related works from different MONID consortia are given by, e.g. [11]–[19]

Extensive research is done in the area of scientific workflows, which enable compute- and data-intensive applications. While analyzing the implementation perspective of our use case, the adaption of Apache Airflow by scientific applications is not common, however in industrial appli-

[1]LOKI https://www.helmholtz.de/loki-pandemics/

[2]https://webszh.uk-halle.de/monid/?p=113

cations it is widely embraced. We found [29], and [30], related to the work presented here. Their approach is well supported by the FAIR principles with the same intent as the focus of our work, but one subtle difference is their implementation baseline. It is based on Docker executors, whereas in our case the scalable and robust ecosystem of Kubernetes is employed. Two further aspects distinguishing our implementation with the related work are: firstly, the integration of Git-based code repository, through which the Apache Airflow image deployment is automated through continuous-integration and continuous-delivery patterns. Secondly, the output data ingested to the S3-interfaced object store can be versioned and can easily be traced back to the past experiment runs. This implies the storage clients compliant with the S3 interface can be re-used by the resulting output of the pipeline presented in this manuscript.

III. Background

This section provides an overview of the technical areas of concern and the respective technologies in the scope of the presented work.

A. Workflow Management

Forecasting of infectious diseases involves the processing of multiple steps and repeated data access in multiple steps. As swift reactions to infectious disease spread are of uttermost importance, the orchestration and automation of multiple processes are indispensable. This brings us to the requirement of low make-span and the seamless execution and monitoring of such workflows. Therefore, an automated mechanism built on an adequate infrastructure ecosystem is required to realize the scientific pipelines.

In this scenario a workflow management system is required to compose and manage a complex set of tasks and data dependencies. There are several open-source options available in the area of workflows and pipelines. To name a few, there are Apache Airflow [31], Luigi [3], and Apache Nifi [4]. They all provide workflow orchestration and enactment capabilities, but are distinct in some areas. As in our scenario, firstly the workflow tasks should be self-contained, i.e. the tasks invoking applications have different set of dependencies, therefore the task execution in containerized form is of prime concern. Luigi and Apache Nifi provide a limited support of, and integration with, containerization ecosystems. Secondly, the integration with the identity and access management is also not trivial in these solutions. On the other hand, Apache Airflow caters well the containerization and the integration with the state-of-the-art identity management systems, it provides an extensible mechanism to dispatch tasks as containers, and can integrate with the container orchestration environments. Apache Airflow is mainly an open-source workflow framework that allows the execution, monitoring, and authoring of distributed workflows as DAGs (Distributed Acyclic Graphs), so called pipelines. The framework can manage workflow tasks in a batch-oriented manner, and schedule the underlying tasks on atomic, user-defined, and frequency-based intervals.

Apache Airflow is based on the “Workflows as code” methodology where users can author pipelines as Python code, and with that it provides a flexible baseline of extensible workflows. The workflow code can be hosted on a version control repository, such as Git. In this manner it can directly be exported to the Apache Airflow service, and instantly visible to its web user interface (UI).

The main elements of the Apache Airflow architecture are as follows. A ***DAG folder***: host a set of DAGs, which can be a directory hosted on a remote file system or version control. A ***scheduler***: which schedules a batch workload of individual tasks within a DAG. An ***executor***: runs a task on executors - act as a tool to process tasks. In our implementation, a Kubernetes executor is configured to run tasks as a pod in a Kubernetes-managed environment. A ***webserver***: provides a web interface to end users. More details of the core concept can be found in the documentation[5].

B. Container Orchestration

In order to host a workflow platform wherein the tasks are supposed to be running as containers, a scalable container orchestration environment is required to facilitate the management, scheduling, and monitoring multiple containers. Docker Swarm[6] and Kubernetes[7] are found to be the most relevant options. Docker Swarm supports containerized workloads similar to Kubernetes, but is particularly well-suited for deployments within the Docker-based ecosystem. In our case, where interoperable container execution is a concern, it is important not to be bound to one specific container runtime. Kubernetes excels in accommodating a wide range of container formats, is capable of handling dynamic workloads at scale. In our research, Kubernetes has proven to be a feasible alternative for our scenario in automating and deploying workflows. Furthermore, it offers broader community support, interoperability, and a vibrant ecosystem of services including scheduling, load balancing, service discovery, and storage mounts.

C. Storage Service

In the realm of modern cloud-enabled data management and storage, the Simple Storage Service (S3) has emerged as a common solution, providing scalable, durable, and highly available object storage. S3, developed by Amazon Web Services (AWS), has become the de facto standard for cloud-based storage, facilitating seamless storage and retrieval of vast amounts of data. MinIO is a notable player in the open-source landscape that has redefined S3 compatibility. It is an object storage server designed for cloud-native and containerized environments, and provides

[3]Luigi https://luigi.readthedocs.io/
[4]Nifi https://nifi.apache.org/
[5]https://airflow.apache.org/docs/apache-airflow/stable/core-concepts/
[6]Docker Swarm https://docs.docker.com/swarm/
[7]Kubernetes https://kubernetes.io

an open source alternative to proprietary S3 solutions supporting large scale data lake and database workloads. MinIO enables organizations to take advantage of the benefits of S3 without being locked into a specific cloud vendor.

IV. Architectural Deployment

This section gives an overview of the integrated deployment and platform architecture. As shown in Fig 1, the architecture is composed of three layers. From top to bottom, the front end or the user facing tier of the deployment encompasses a user domain. In this tier, the main roles are data engineer and end user. The data engineer's role is to setup and configure workflow site and push environment-specific Airflow container image to the container registry. The end user is the domain scientist which is responsible to author and publish DAGs to the DagBag repository. For this setup, a Git repository is used to host DAGs, and this is enabled by using Apache Airflow's ***git-sync*** feature. In order to access the site, the end user has to be known to the site's identity provider. The access control is supported through an OpenID Connect (OIDC)[8]-compliant service, called Unity IdM[9]. Further in this direction, the Apache Airflow's security model follows the OAuth 2.0 Authorization Framework's Authorization Code Grant [10] to grant access to the users.

As soon as the end user is successfully authenticated and authorized, he is taken to the Airflow's web user interface that interacts with the workflow management tier. On this layer, the interface between end user and the workflow management service - Apache Airflow is established to run and manage remote DAG tasks. From the deployment perspective, Apache Airflow provides several alternatives, namely, stand alone, distributed, and containerized. In our approach the containerized option is used. It is because the container orchestration engines implicitly manage the life cycle of containers without exposing internal intricacies to service users. Given that, the Apache Airflow instance is deployed on an on-premise Kubernetes Management system called Rancher. Rancher provides a KaaS (Kubernetes as a Service) platform for hosting multi-tenant Kubernetes instances. In our scenario, a typical end user interaction begins with a request to a load balancing server called NGINX (Ingress-based). This server takes the request to the web UI, where end user sees the DAGs stored on Git repository. The scheduled DAGs or tasks are managed by the Scheduler, which then forwards the incoming request to the Executor. The Executor is responsible for running the tasks of a DAG. In our case, Apache Airflow's Kubernetes Executor is used, which facilitates DAG tasks as individual Pods[11], so called TaskPods. The executor can be configured to have tasks managed as a local Airflow managed executor, while considering a deployment within container orchestration KubernetesExecutor environment seems to be the optimal choice. It is because the KubernetesExecutor is integrated within the environment and can better delegate the job of spawning, re-scheduling and suspending tasks to the Kubernetes' scheduler.

Managing such an infrastructure mandates careful monitoring and alerting of deployed services. In our deployment we are using Prometheus[12] for aggregating the monitoring and alerting data. As it is well suited for containerized environments, the integration and support is favorable to our deployment. The data aggregation and summarising is one part, the visualization gives a visual insight. To visualize the gathered data and serve this purpose Grafana is used. Apache Airflow does not directly support Prometheus, but rather exports metric data to statsd-exporter[13] and then to the Prometheus endpoint. Fig. 1 shows the interconnect of the monitoring and workflow management sub-systems.

In the Kubernetes environment, the deployment of applications such as workflow management system and respective data stores, the associated Pods as deployment units require to be configured with confidential tokens, for instance, username and password or certificates. In such cases, Kubernetes' Secret objects are utilized and can directly be attached to the deployment descriptor. In our approach the Secret management is automated by fetching the required Kubernetes Secrets from the JSC infrastructure's centrally managed HashiCorp's Vault[14] instance. The presented architecture utilizes a myriad of underlying services provided by the native Kubernetes environment. As this description goes beyond the scope of this paper, we refer to the Kubernetes documentation for more information. The lowest layer of our implementation is based on the cloud hosting layer, which uses Openstack[15] as an Infrastructure-as-a-Service (IaaS), called JSC-Cloud[16].

V. Scientific Use Case

We use the presented architecture for nowcasting the number of mild infections, hospitalized patients, patients in intensive care, and deaths as returned by the model proposed in [32]. Specifically, the model is based on hybrid graph-ordinary differential equations, in order to include travel and commuting activities; see [33]. The model applies a metapopulation strategy and constructs one ODE-SECIR-type model for each county in Germany, which are then connected via a multi-edge graph to allow for exchange of local populations. The local ODE-SECIR-type model was extended to include vaccinations and reinfections in [32].

We now describe our workflow library *MEmiliflow*, available at [34]. MEmiliflow is designed as the connecting piece between MEmilio [23] and Airflow, and contains several useful CLI-scripts in its pipeline subdirectory. Each

[8]OpenID Chttps://openid.net

[9]Unity IdM https://unity-idm.eu

[10]OAuth 2.0 https://datatracker.ietf.org/doc/html/rfc6749#section-4.1

[11]Pod: Pod is the basic abstraction that encapsulates one or more containers.

[12]Prometheus https://prometheus.io/

[13]statsd-exporter https://github.com/prometheus/statsd_exporter

[14]HashiCorp Vault https://www.vaultproject.io/

[15]Openstack https://www.openstack.org/

[16]JSC-Cloud https://cloud.jsc.fz-juelich.de/

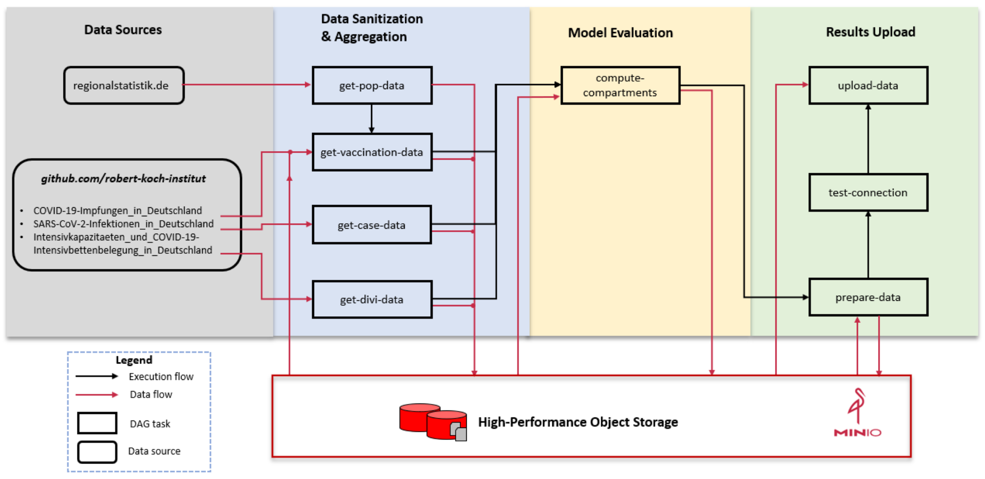
Data Sources
Data Sanitization & Aggregation
Model Evaluation
Results Upload
regionalstatistik.de
github.com/robert-koch-institut
COVID-19-Impfungen_in_Deutschland
SARS-CoV-2-Infektionen_in_Deutschland
Intensivkapazitaeten_und_COVID-19-Intensivbettenbelegung_in_Deutschland
get-pop-data
get-vaccination-data
get-case-data
get-divi-data
compute-compartments
upload-data
test-connection
prepare-data
High-Performance Object Storage
MINIO
Legend
Execution flow
Data flow
DAG task
Data source

Fig. 2. Visualization of the execution and data flows for our nowcasting DAG *dag_loki_ESID_datapipeline.py*.

workflow step, i.e. DAG task in Figure 2, has a corresponding CLIscript, that also takes care of data transfers to and from S3. Furthermore, we design each workflow step to be idempotent to enable easier repetition of failing steps and debugging. The data downloaded from public sources and the produced (intermediate and final) data are stored in compressed form in S3 for archiving purposes. We use GitLab Ops to automatically build and provide docker images of MEmiliflow.

Next, we describe our DAG for nowcasting, available at [35]. Figure 2 details the DAGs' steps, each one mapping to a CLIscript in MEmiliflow. On each day, Airflow automatically downloads the available public data on reported infections and uses the initialization strategy described by [32] to estimate the total number of individuals currently undergoing, e.g. a mild infection or a hospitalization. Because the reported data is uploaded with a delay of one day, our estimations refer to the previous day. We then run the model with ranges for all the different model parameters that have been obtained from the literature and previous study results [32]. The output of the model is written to the S3-compliant MinIO instance. The final steps are repackaging the data to a format compatible with ESID's database and finally the upload to the ESID [24] web application running at [36]. Akin to continuous delivery, the DAG automatically fetches the latest MEmiliflow's docker image from the Container Registry, as soon as code changes are applied to the image descriptor (Dockerfile).

In the future, we will extend our scientific use case in three ways. Firstly, we are currently working towards including forecasts, i.e. adding predictions into the near future using the model. Secondly, we aim to add automatic parameter estimation and updates based on the recent time series of infected individuals and hospitalizations. Thirdly, we are working on updating the pipeline to allow for variable model selection to include different levels of granularity as well as important aspects such individual behavior, stochasticity, or waning immunity. Including these updates into our presented architecture by creating additional DAGs is technically straightforward.

VI. Conclusion

With the widespread distribution of vaccines on the global scale, the share of severe or critical infections has largely declined. Consequently, the demand and need for automatic nowcasting of COVID-19 has naturally declined too. However, the SARS-CoV-2 pandemic has clearly demonstrated the benefit of digital tools for decision makers, public health experts and local health authorities to react in a swift and suitably adapted manner. As the frequency of pandemics is expected to increase [37], automatic workflows have to be set up to be prepared for any potential future infectious diseases that might emerge.

Inspired by the FAIR and FAIR4RS principles, we presented a multi-tier architecture to accommodate the deployment of a dynamically provisioned workflow management system in a containerized ecosystem. Specifically, our architecture consisting of open-source components with permissive or copy-left licenses, is implementable by third-parties and our MinIO bucket has public readonly access. It can furthermore be used as a reference model for case studies required to run DAGs on Kubernetes-based environments. Our scientific case study on COVID-19 is demonstrated with real datasets and state-of-the-art models to validate our approach. As a future direction, we foresee two enhancements. Firstly, compute- and memory-intensive tasks will be ported to high-performance computing (HPC) resources. Therefore, it requires enhancing Apache Airflow's DAG interoperability with HPC endpoints. Secondly, the current Apache Airflow implementation cannot manage multi-tenancy in its entirety. The current implementation will be adjusted to expand multiple user groups with fine-grained access control.

Acknowledgment

This work was performed as part of the project 'Integrated Early Warning System for Local Recognition, Prevention, and Control for Epidemic Outbreaks' (LOKI), which is funded by the Initiative and Networking Fund of the Helmholtz Association (grant agreement number KA1-Co-08). The authors HZ and MJK have partially been funded by German Federal Ministry for Digital and Transport under grant agreement FKZ19F2211A, project PANDEMOS.

References

[1] World Health Organization, "Coronavirus disease (COVID-19): Herd immunity, lockdowns and COVID-19," 2020. [Online]. Available: https://www.who.int/news-room/q-a-detail/herd-immunity-lockdowns-and-covid-19

[2] World Health Organization Team Data, Analytics & Delivery, "World health statistics 2023: monitoring health for the SDGs, sustainable development goals," World Health Organization, Geneva, Tech. Rep., 2023. [Online]. Available: https://www.who.int/publications/i/item/9789240074323

[3] F. Dorn, S. Khailaie, M. Stoeckli, S. C. Binder, T. Mitra, B. Lange, S. Lautenbacher, and others, "The common interests of health protection and the economy: evidence from scenario calculations of COVID-19 containment policies," *The European Journal of Health Economics*, vol. 24, no. 1, pp. 67–74, Feb. 2023.

[4] J. M. Brauner, S. Mindermann, M. Sharma, D. Johnston, J. Salvatier, T. Gavenčiak, and others, "Inferring the effectiveness of government interventions against COVID-19," *Science*, vol. 371, no. 6531, p. eabd9338, Feb. 2021.

[5] M. an der Heiden, A. Hicketier, and V. Bremer, "Wirksamkeit und Wirkung von anti-epidemischen Maßnahmen auf die COVID-19-Pandemie in Deutschland (StopptCOVID- Studie)," Tech. Rep., Jul. 2023.

[6] C. C. Kerr, R. M. Stuart, D. Mistry, R. G. Abeysuriya, K. Rosenfeld, G. R. Hart, and others, "Covasim: An agent-based model of COVID-19 dynamics and interventions," *PLOS Computational Biology*, vol. 17, no. 7, pp. 1–32, Jul. 2021.

[7] M. J. Keeling, M. J. Tildesley, B. D. Atkins, B. Penman, E. Southall, G. Guyver-Fletcher, A. Holmes, H. McKimm, E. E. Gorsich, E. M. Hill, and others, "The impact of school reopening on the spread of COVID-19 in England," *Philosophical Transactions of the Royal Society B*, vol. 376, no. 1829, p. 20200261, 2021.

[8] X. Liu, J. Huang, C. Li, Y. Zhao, D. Wang, Z. Huang, and K. Yang, "The role of seasonality in the spread of COVID-19 pandemic," *Environmental Research*, vol. 195, p. 110874, 2021.

[9] F. S. Lobato, G. B. Libotte, and G. M. Platt, "Mathematical modelling of the second wave of COVID-19 infections using deterministic and stochastic SIDR models," *Nonlinear Dynamics*, vol. 106, no. 2, pp. 1359–1373, 2021.

[10] S. Sturniolo, W. Waites, T. Colbourn, D. Manheim, and J. Panovska-Griffiths, "Testing, tracing and isolation in compartmental models," *PLOS Computational Biology*, vol. 17, no. 3, pp. 1–28, Mar. 2021.

[11] M. J. Kühn, D. Abele, S. Binder, K. Rack, M. Klitz, J. Kleinert, and others, "Regional opening strategies with commuter testing and containment of new SARS-CoV-2 variants in Germany," *BMC Infectious Diseases*, vol. 22, no. 1, p. 333, Dec. 2022.

[12] S. A. Müller, M. Balmer, W. Charlton, R. Ewert, A. Neumann, C. Rakow, T. Schlenther, and K. Nagel, "Predicting the effects of COVID-19 related interventions in urban settings by combining activity-based modelling, agent-based simulation, and mobile phone data," *PLOS ONE*, vol. 16, no. 10, p. e0259037, Oct. 2021.

[13] L. Tapp, V. Kurchyna, F. Nogatz, J. O. Berndt, and I. J. Timm, "School's out? simulating schooling strategies during COVID-19," in *International Workshop on Multi-Agent Systems and Agent-Based Simulation*. Springer, 2022, pp. 95–106.

[14] S. Bauer, S. Contreras, J. Dehning, M. Linden, E. Iftekhar, S. B. Mohr, and others, "Relaxing restrictions at the pace of vaccination increases freedom and guards against further COVID-19 waves," *PLOS Comp Bio*, vol. 17, no. 9, p. e1009288, Sep. 2021.

[15] L. Schüler, J. M. Calabrese, and S. Attinger, "Data driven high resolution modeling and spatial analyses of the COVID-19 pandemic in Germany," *PLOS ONE*, vol. 16, no. 8, p. e0254660, Aug. 2021.

[16] M. V. Barbarossa, J. Fuhrmann, J. H. Meinke, S. Krieg, H. V. Varma, N. Castelletti, and T. Lippert, "Modeling the spread of COVID-19 in Germany: Early assessment and possible scenarios," *PLOS ONE*, vol. 15, no. 9, pp. 1–22, Sep. 2020.

[17] I. Rodiah, P. Vanella, A. Kuhlmann, V. K. Jaeger, M. Harries, G. Krause, and others, "Age-specific contribution of contacts to transmission of SARS-CoV-2 in Germany," *European Journal of Epidemiology*, vol. 38, no. 1, pp. 39–58, Jan. 2023.

[18] J. Mohring, M. Burger, R. Feßler, J. Fiedler, N. Leithäuser, J. Schneider, M. Speckert, and J. Wlazlo, "Starker Effekt von Schnelltests (Strong effect of rapid tests)," Apr. 2023. [Online]. Available: http://arxiv.org/abs/2304.05938

[19] Y. Kheifetz, H. Kirsten, and M. Scholz, "On the Parametrization of Epidemiologic Models—Lessons from Modelling COVID-19 Epidemic," *Viruses*, vol. 14, no. 7, p. 1468, Jul. 2022.

[20] S. Memon, G. Cavallaro, B. Hagemeier, M. Riedel, and H. Neukirchen, "Automated Analysis of Remotely Sensed Images Using the UNICORE Workflow Management System," in *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, July 2018, pp. 1128–1131, dOI:10.1109/IGARSS.2018.8519364.

[21] M. Barker, N. P. Chue Hong, D. S. Katz, A.-L. Lamprecht, C. Martinez-Ortiz, F. Psomopoulos, and others, "Introducing the FAIR Principles for research software," *Scientific Data*, vol. 9, no. 1, p. 622, Oct. 2022.

[22] C. de Visser, L. F. Johansson, P. Kulkarni, H. Mei, P. Neerincx, K. Joeri van der Velde, P. Horvatovich, A. J. van Gool, M. A. Swertz, P. A. C. t. Hoen, and A. Niehues, "Ten quick tips for building fair workflows," *PLOS Computational Biology*, vol. 19, no. 9, pp. 1–13, 09 2023.

[23] M. J. Kühn, D. Abele, D. Kerkmann, S. A. Korf, H. Zunker, A. C. Wendler, J. Bicker, D. K. Nguyen, and others, "Memilio v1.0.0 - a high performance modular epidemics simulation software," December 2023. [Online]. Available: https://elib.dlr.de/201660/

[24] P. K. Betz, J. Stoll, V. Grappendorf, J. Gilg, M. Zeumer, M. Klitz, and others, "ESID: Exploring the design and development of a visual analytics tool for epidemiological emergencies," in *2023 IEEE VIS Workshop on Visualization for Pandemic and Emergency Responses (Vis4PandEmRes)*, 2023, pp. 8–14.

[25] J. Bracher *et al.*, "National and subnational short-term forecasting of COVID-19 in germany and poland during early 2021," *Communications Medicine*, vol. 2, no. 1, p. 136, Oct 2022.

[26] D. Wolffram *et al.*, "Collaborative nowcasting of covid-19 hospitalization incidences in germany," *PLOS Computational Biology*, vol. 19, no. 8, p. e1011394, Aug. 2023.

[27] F. Bergström, F. Günther, M. Höhle, and T. Britton, "Bayesian nowcasting with leading indicators applied to COVID-19 fatalities in sweden," *PLOS Computational Biology*, vol. 18, no. 12, p. e1010767, Dec. 2022.

[28] S. K. Greene *et al.*, "Nowcasting for real-time COVID-19 tracking in new york city: An evaluation using reportable disease data from early in the pandemic," *JMIR Public Health Surveill*, vol. 7, no. 1, p. e25538, Jan 2021.

[29] F. M. Mione, A. N. Silva, M. F. Luna, M. N. Cruz B., and E. C. Martinez, "Managing experimental-computational workflows in robotic platforms using directed acyclic graphs," in *14th International Symposium on Process Systems Engineering*, ser. Computer Aided Chemical Engineering, Y. Yamashita and M. Kano, Eds. Elsevier, 2022, vol. 49, pp. 1495–1500.

[30] L. Kaspersetz, F. Schröder-Kleeberg, F. M. Mione, E. C. Martinez, P. Neubauer, and M. N. Cruz-Bournazou, "Automation of experimental workflows for high throughput robotic cultivations," *bioRxiv*, 2023.

[31] B. P. Harenslak, J. R. de Ruiter, and J. Brierley, *Data Pipelines with Apache Airflow*. Manning Publications Co., 2022.

[32] W. Koslow, M. J. Kühn, S. Binder, M. Klitz, D. Abele, A. Basermann, and M. Meyer-Hermann, "Appropriate relaxation of non-pharmaceutical interventions minimizes the risk of a resurgence in SARS-CoV-2 infections in spite of the Delta variant," *PLOS Computational Biology*, vol. 18, no. 5, p. e1010054, May 2022.

[33] M. J. Kühn, D. Abele, T. Mitra, W. Koslow, M. Abedi, K. Rack, and others, "Assessment of effective mitigation and prediction of the spread of SARS-CoV-2 in Germany using demographic information and spatial resolution," *Mathematical Biosciences*, vol. 339, p. 108648, Sep. 2021.

[34] "MEmiliflow," https://codebase.helmholtz.cloud/loki/memiliflow.

[35] "DAGBag," https://codebase.helmholtz.cloud/loki/dagbag.

[36] "ESID preview," https://zam10063.zam.kfa-juelich.de/development-preview/.

[37] IPBES, "Workshop Report on Biodiversity and Pandemics of the Intergovernmental Platform on Biodiversity and Ecosystem Services (IPBES)," Zenodo, Tech. Rep., Oct. 2020, version Number: 1.4.