Smart Contracts and Decentralized Finance

Parametric Weather Insurance

submitted by:

Kai Frehner, Benjamin Gerber

Matrikel–Nr.: 20-647-970, 21-550-579

supervised by:

Prof. Dr. Fabian Schär

Place, Date:

Basel, 21.11.2025

Contents

1 Introduction

Parametric insurance has become one of the most dynamic innovations in the insurance industry. Unlike traditional indemnity products, where the payout depends on an assessed loss, parametric insurance compensates the policyholder as soon as a predefined, objective trigger is reached. This structure eliminates lengthy loss adjustment processes, increases transparency, and allows for flexible product design across various lines of business such as agriculture, natural catastrophe, and travel insurance.

A major technological driver behind the rise of parametric solutions is the use of smart contracts. As self-executing programs on a blockchain, smart contracts automatically enforce contractual rules once the relevant conditions are fulfilled. Their deterministic execution, immutability, and auditability make them well-suited for parametric insurance, where payouts rely on measurable external data such as precipitation, wind speed, or flight delays. By integrating smart contracts with reliable oracle systems, the entire claims process, from data retrieval to payout, can be automated. This reduces administrative effort, prevents disputes, and enables fast, predictable settlements for policyholders.

This paper outlines the core principles of parametric insurance and contrasts them with traditional indemnity structures. We then present a real-world example from Baloise's multi-parametric travel insurance product, focusing on the bad-weather component developed with the German start-up Wetterheld. The central part of the project is a simplified implementation of a parametric weather insurance product using a smart contract on the Ethereum Sepolia test network, illustrating how blockchain technology can enhance transparency, automation, and efficiency in parametric insurance.

2 Parametric Insurance

In the context of non-life insurance, the intuitive idea is that a predefined risk is covered by an insurer who indemnifies the policyholder in the event of a loss. The indemnity payment is directly linked to the actual loss incurred by the insured. However, the landscape of non-life insurance products encompasses a much broader variety of contract types than this classical form of indemnity insurance. Figure 1 illustrates the different categories of indemnity and non-indemnity insurance contracts:

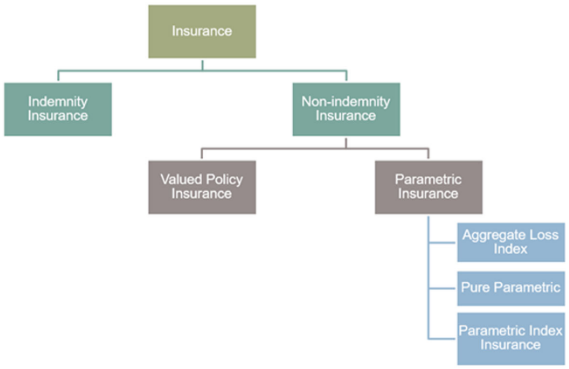

Figure 1: Classification of nonlife contracts [2]

In the following, we refer to "Pure Parametric Contracts" simply as *Parametric Insurance*. This class of contracts provides a predefined payout to the insured once a specific trigger event occurs, irrespective of the actual monetary loss. Typical trigger variables include, for example, earthquake magnitude, wind speed, or flight delay. Table 1 summarises key differences between traditional indemnity insurance and parametric solutions.

	Traditional insurance	**Parametric insurance**
Recovery	Reimbursement of ajusted loss	Pre-agreed payment structure based on a event parameter (often binary)
Trigger	Payment triggered by actual loss or damage	Payment triggered by an event occurrence exceeding the parametric threshold
Loss assement and payment	Complex and based on loss adjuster assesment	Transparent, predictable, based on a parameter, quick settlement
Structure	Standard product and contract wordings	Tailored product with high structuring flexibility

Table 1: Traditional versus parametric insurance [3]

These structural characteristics make parametric solutions particularly attractive for several lines of business, such as agriculture, natural catastrophe insurance, and travel insurance [3]. In recent years, parametric insurance has been one of the fastest-growing segments in the insurance market, with premium volumes expected to triple during the 2020s, reaching almost USD 30 billion [4]. Despite this rapid growth, parametric products remain a niche market compared to the approximately USD 7 trillion in global insurance premiums [5].

2.1 Baloise and Wetterheld - A parametric travel insurance

In 2023, the Swiss insurer Baloise was awarded second place at the Swiss Insurance Innovation Award for its multi-parametric travel insurance solution, *Parasurance*. This new parametric insurance product allows policyholders to select coverage for flight delay, luggage delay, adverse weather, or any combination thereof during their trip. Additionally, motor third-party liability insurance

can be added. The parametric bad-weather component is offered in collaboration with the German start-up Wetterheld. In this parametric coverage, the trigger is activated if the precipitation at a predetermined destination exceeds a specified threshold (in millimetres) within a given time period. While Baloise provides the insurance product, Wetterheld is responsible for monitoring and verifying the relevant parameters. Note that although *Parasurance* covers multiple risks, this section focuses exclusively on the bad-weather component [6].

Time restriction To mitigate moral hazard, parametric weather insurance cannot be purchased later than 14 days prior to the trip's commencement. Forecast accuracy declines sharply beyond two weeks, rendering longer-term forecasts no more reliable than random guessing. The duration of trips eligible for bad-weather coverage ranges from 2 to 92 days.

Geographical scope Policyholders must provide the GPS coordinates of their travel destination. Wetterheld verifies eligibility for compensation using publicly accessible weather data from Meteostat [7], based on precipitation measurements at the reported coordinates. Coverage is restricted to the specified location; changes of destination during the trip are not permitted. Only trips within Europe are eligible for this insurance.

Definition of bad weather The coverage is triggered if daily precipitation (including rainfall, snow, or hail) exceeds 2.9 mm, considering only the period between 10 a.m. and 6 p.m. A precipitation amount of 2.9 mm corresponds approximately to 90 minutes of light rain or 10–15 minutes of heavy rain [8].

Deductible days The coverage also accounts for regional differences in expected precipitation. The likelihood of adverse weather varies, e.g., between the United Kingdom and Andalusia. Wetterheld therefore applies *deductible days*: for a seven-day trip to Cardiff, the first three days may not be covered, whereas for a trip to Sevilla only the first day is excluded. Moreover, the number of days with significant precipitation varies throughout the calendar year.

Pricing Determining an appropriate risk premium is the most critical aspect and remains confidential. While specific modelling and pricing details are not publicly available, the pay-out trigger remains fully transparent.

3 Implementation of a parametric weather insurance on a Smart Contract

In this section the implementation of a parametric weather insurance on a Smart Contract is discussed. First a brief summary of Smart Contracts is provided.

3.1 Basics of Smart Contracts

A smart contract is a self-executing program stored on a blockchain that automatically enforces and carries out predefined rules once specific conditions are met. Its logic is executed deterministically by all validating nodes in the network, ensuring identical outcomes without relying on intermediaries or trust between contracting parties. Because all state transitions and transactions are recorded on an immutable distributed ledger, smart contracts offer transparency, auditability, and resistance to tampering.

Technically, a smart contract is deployed as bytecode at a dedicated blockchain address and can be invoked through transactions that trigger state changes. The contract used in this study is implemented in Solidity and deployed on the *Sepolia* Ethereum test network, a low-cost proof-of-stake testnet that reproduces the execution environment of the Ethereum mainnet. This allows realistic testing of contract logic—such as fund transfers, threshold evaluations, and automated payouts—without financial exposure.

Smart contracts are frequently used to automate processes in decentralized applications, including parametric insurance. Because blockchains cannot natively access external data, such contracts rely on oracle mechanisms that deliver off-chain information (e.g., weather measurements) to the chain. Once this data is received, the contract evaluates it according to predefined criteria and deterministically executes the corresponding outcome, such as triggering a payout.

3.2 Simplified Implementation of parametric weather insurance

For the project in "Smart Contracts and Decentralised Finance" the product presented was partially implemented in a smart contract on the Sepolia-ETH test net. The code was programmed using the programming language Solidity and the development environment used was the REMIX browser IDE. Many aspects were simplified and some were out of the scope of our project. When a topic was considered out of scope, the theoretical solution is outlines as well as the simplified implemented version are specified.

Time restriction The time restrictions implemented in the smart contract mirror those used in the Baloise insurance product. The start and end dates of a policy are validated using Solidity's `require` statements. For testing purposes, the minimum waiting period of 14 days has been removed.

Geographical scope In this implementation the insured would provide a start and an end date of their trip as well as gps coordinates of their location. The location is then used to calculate the average number of days in a year with a precipitation larger than 2.9mm using R. For each 0.5° x 0.5° grid cell, there is on Copernicus, which is a open source program by the European Comission, for historical weather data for drought, flood, temperature, precipitation etc. available. This might be the source to verify whether the trigger is hit.

Definition of bad weather Bad weather is defined by 2.9mm precipitation within a time period between 8 AM and 6 PM per day in accordance with the Baloise product.

Deductible days The methodology underlying the calculation of deductible days provided by Wetterheld is not transparently available to customers. For this project, we developed our own approach, using five risk categories (A–E) to determine the number of deductible days for a given location. We utilized data from the European Climate Assessment & Dataset (ECA&D) project, which provides daily climate records from over 16,000 weather stations, often dating back to the early 20th century [10]. To establish the thresholds for our risk categories, we analyzed the annual number of rainy days (defined as days with precipitation equal to or exceeding 2.9 mm) at 1,000 randomly selected weather stations. Using all stations would have resulted in datasets that were too large and beyond the scope of this project. Each of the five risk categories corresponds to a 20-percent quantile of the annual number of rainy days within the sampled dataset (Calculations and Plotting done using R, see file risk_category.R):

Our risk categories are therefore defined as follows:

A	**B**	**C**	**D**	**E**
< 50	$50 \leq x < 64$	$64 \leq x < 77$	$77 \leq x < 93$	$93 \leq x$

Table 2: Risk categories based on quantiles of number of annual rainy days

For instance, if the average annual number of rainy days at a given location is 62, the corresponding risk category would be "B." The number of deductible days is subsequently determined using the specified formula:

$$D_{\text{ded}} = \text{round}\big(D_{\text{tot}} \cdot \mu_X\big)$$

where: D_{ded} = Deductible Days, D_{tot} = Duration of the trip, μ_X = risk factor, with $\mu_A = 0.1, \mu_B = 0.2, \ldots, \mu_E = 0.5$

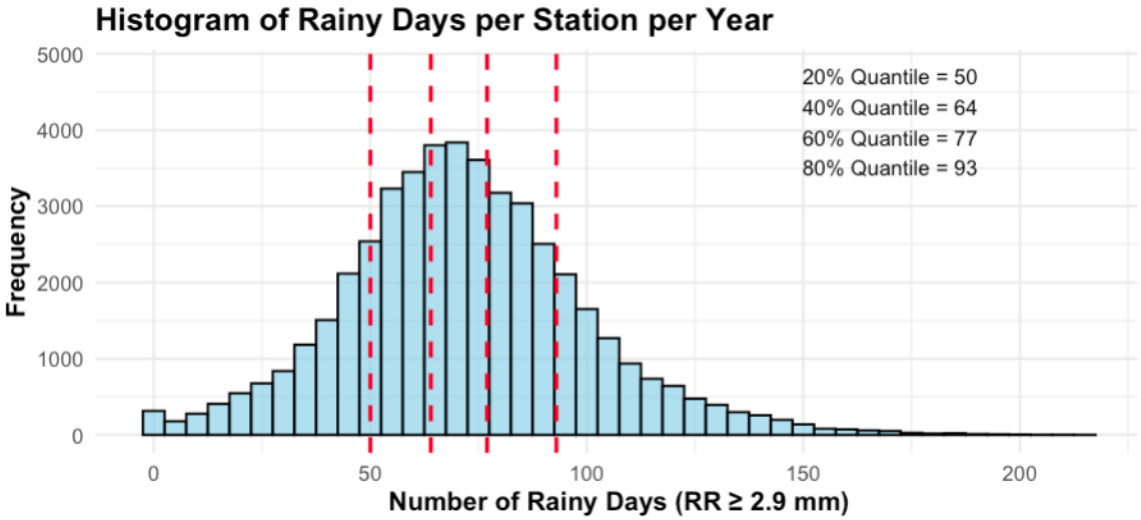

Figure 2: Quantiles of annual rainy days

Note that in this initial implementation the monthly change in precipitation for a specific location is not considered.

Pricing Due to the proprietary nature of the Baloise product, the exact pricing methodology is not publicly disclosed and is therefore considered out of scope for this project. For testing purposes, a simplified and deliberately rudimentary premium formula is employed. This approximation derives the premium from the duration of the insured trip, the level of coverage, the number of deductible days, and a coarse estimate of the likelihood of rain in the insured region. In addition, a safety margin of $\Phi = \frac{11}{10}$ is included to account for uncertainty.

$$\text{premium} = \text{coverage} \cdot (D_{\text{tot}} - D_{\text{ded}}) \cdot \mu_X \cdot \Phi$$

Example Berlin in September 2025 In order to illustrate a parametric weather insurance, we assume a trip to Berlin between 9 September and 17 September 2025 (days 252–260 in Figure 3). During this period, precipitation exceeds the threshold of 2.9 mm on four days. Berlin is assigned to risk category "B". Therefore, the number of deductible days is

$$D_{\text{ded}} = \text{round}\big(8 \cdot 0.2\big) = 2$$

With the number of bad-weather days = 4, the policy would have triggered a payout of

$$(4 - 2) \cdot \text{coverage}.$$

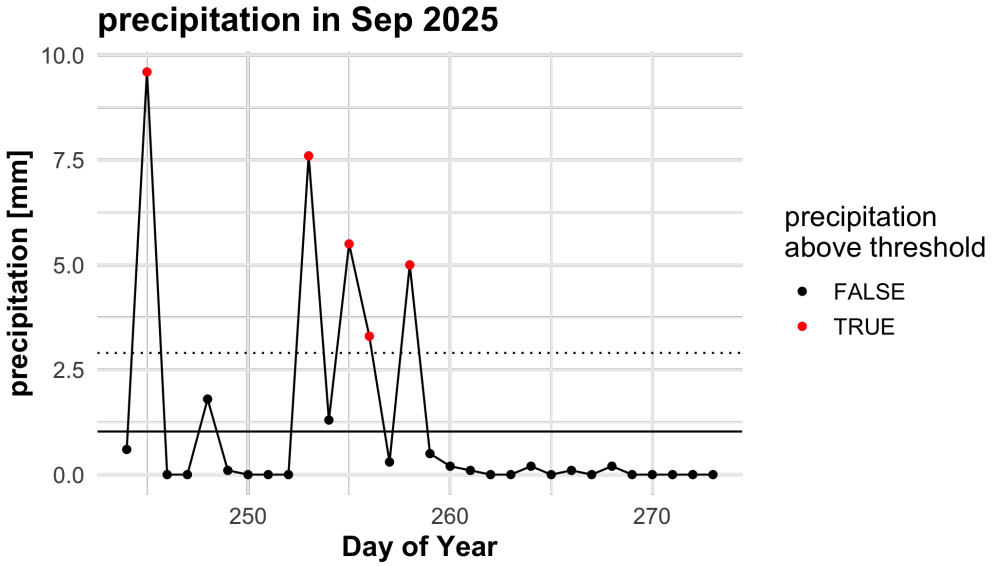

Figure 3: example of a triggered parametric weather insurance
(Calculations and Plotting done using R, see file example_plotting.R):

3.3 Specifics of the Smart Contract

This section describes the implementation of the parametric weather insurance using a smart contract. Some features outlined above are simplified, either because they fall outside the scope of this project or because they proved impractical for testing purposes.

The basic structure of the smart contract is that the insurer deploys the contract, and users can purchase policies directly on-chain. Policy information (i.e., policyholder, region of travel, number of deductible days, and the threshold for bad weather (2.9 mm)) is stored within the contract. Aggregated weather data for the relevant time period is imported once daily, which triggers the function that checks which policies are affected. This function is described in more detail below.

Input Checks Because the contract requires user interaction when setting up a policy, all inputs undergo validation. This is primarily achieved using Solidity's `require` statements, which ensure that user-provided data is valid and that the premium payment meets the required amount.

Events Key state changes in the smart contract are recorded using events. These events are written to the transaction logs, providing an auditable record of contract activity. In this implementation, events are emitted when a new policy is created, when new weather data is imported, and when policies are affected by an incoming weather data point. Additionally, the contract emits an event when a policy expires and a payout is executed.

Data Import To supply the contract with external information, an oracle mechanism is used. A blockchain oracle is an external service that delivers off-chain data—such as weather measurements—to a smart contract, enabling it to react to real-world events that the blockchain cannot observe natively. As noted by Antonopoulos and Wood (2018), oracles are essential for linking deterministic on-chain logic with unpredictable off-chain inputs. In this implementation, weather data is imported manually through a function call, which enables rapid testing by allowing new data points to be added within seconds rather than waiting for real-world observations.

Outline of the `trigger` Function The core logic of the parametric insurance mechanism is implemented in the `trigger` function, which is executed whenever a new weather data point is uploaded. Each time the function runs, it iterates through all active policies and evaluates their status (active, not yet active, or expired). It then determines whether the new data point qualifies as a bad-weather day for the relevant region. If so, the policy is updated accordingly, and the appropriate events are emitted.

Once a policy has expired, the function checks whether the number of bad-weather days exceeds the deductible. If the threshold is met, the payout is computed as

$$(\text{bad-weather days} - \text{deductible days}) \cdot \text{coverage}.$$

The contract's balance is verified to ensure sufficient funds for the payout. If the payment is successfully processed, the expired policy is removed from the contract state, while all associated event logs remain stored on the blockchain.

Figure 4 provides a visual representation of the `trigger` function.

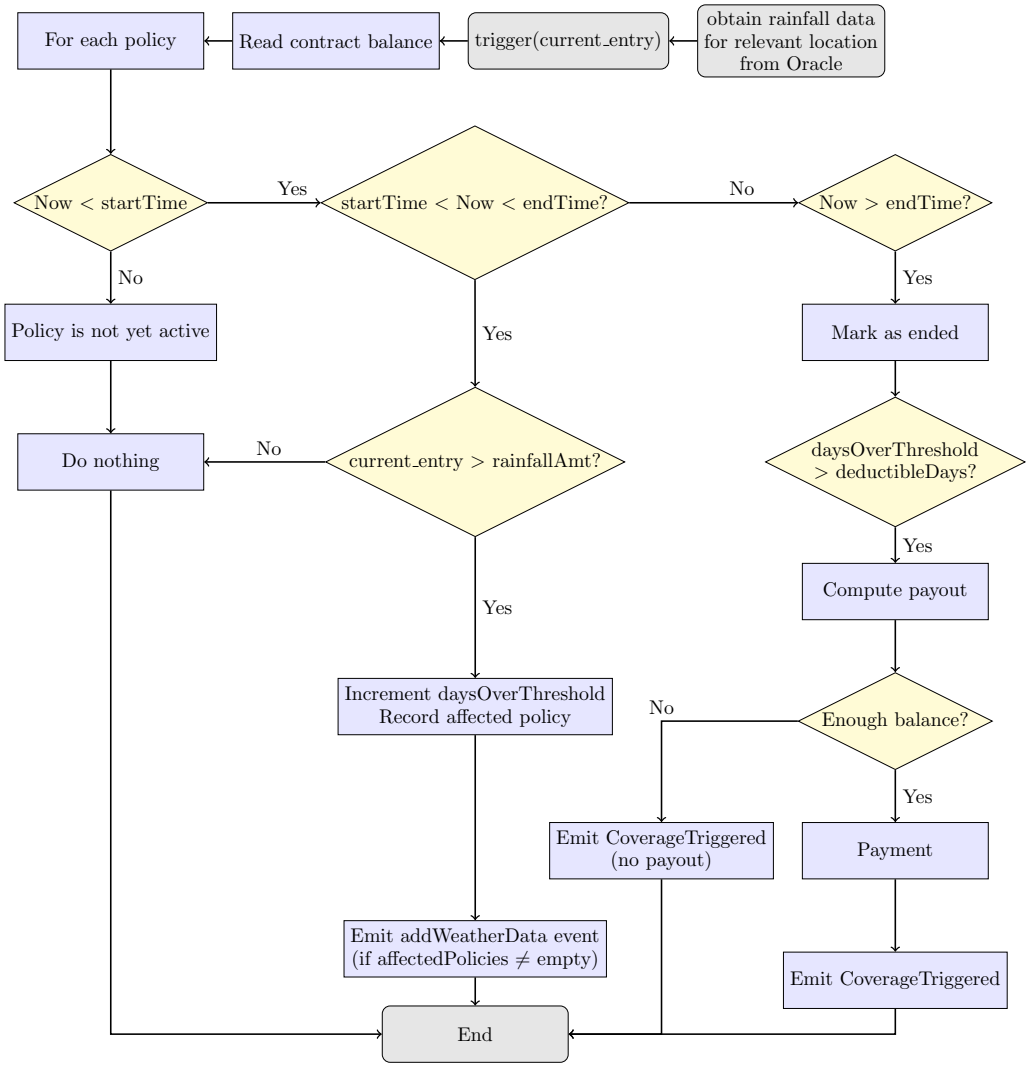

Figure 4: Flowchart of the `trigger` function

4 Conclusion

Parametric insurance offers fast, transparent, and objective payouts based on measurable trigger events instead of traditional loss adjustment. The prototype developed in this project demonstrates how smart contracts can strengthen these advantages through automated execution, immutable documentation, and deterministic evaluation. The implementation on the Ethereum Sepolia test network shows that even simplified versions of parametric weather insurance can be encoded efficiently, including deductible-day logic, regional risk factors, and automated payout calculation.

However, the reliance on oracles introduces notable risks. Since blockchains cannot access external data directly, smart contracts depend on oracle services to supply accurate weather information. Faulty, delayed, or manipulated data can cause incorrect payouts or unjustified claim denials, making the oracle a potential single point of failure. Oracle infrastructure is also vulnerable to cyberattacks or outages, and it shifts part of the trust from the insurer to the data provider. These risks highlight that while smart contracts reduce operational and counterparty risk, they cannot eliminate dependencies on external information sources.

Mitigation strategies—such as decentralised oracle networks, multi-source data aggregation, or cryptographic verification—offer promising improvements but also add complexity. Despite these challenges, the results of this project show that blockchain-based implementations can significantly increase efficiency, transparency, and speed in parametric insurance. As oracle systems mature and become more robust, parametric products on blockchain have strong potential to scale and deliver reliable, user-centric risk transfer solutions.

References

[1] Mack, T. (1993): *Distribution-free calculation of the standard error of chain ladder reserve estimates.* ASTIN Bulletin 23/2, S. 171-183.

[2] Lin, X., Kwon, WJ. (2020): *Application of parametric insurance in principle-compliant and innovative ways.* Risk Manag Insur Rev 23, p. 121-150.

[3] Swiss Re Corporate Solutions, *Comprehensive guide to Parametric Insurance*, https://corporatesolutions.swissre.com/dam/jcr:0cd24f12-ebfb-425a-ab42-0187c241bf4a/2023-01-corso-guide-of-parametric-insurance.pdf. Accessed, November 8, 2025.

[4] Swiss Re Insights, *Parametric insurance – a long history, a bright future*, https://corporatesolutions.swissre.com/insights/knowledge/evolution-of-parametric-insurance.html. Accessed, November 8, 2025.

[5] Allianz, *Allianz Global Insurance Report 2025: Rising demand for protection*, https://www.allianz.com/en/economic_research/insights/publications/specials_fmo/250527-global-insurance-report.html. Accessed, November 8, 2025.

[6] Wetterheld, *Wetterheld Website*, https://wetterheld.com/home/de/. Accessed, November 11, 2025.

[7] Meteostat, *Meteostat webiste*, https://meteostat.net/de/. Accessed, November 10, 2025.

[8] Deutscher Wetterdienst, *Wetterlexikon: Niederschlagsintensität*, https://www.dwd.de/DE/service/lexikon/Functions/glossar.html?lv2=101812&lv3=101906. Accessed, November 16, 2025.

[9] Copernicus, *European Earth observation program*, https://www.copernicus.eu/en. Accessed, November 20, 2025.

[10] ECAD, *European Climate Assessment and Dataset*, https://www.ecad.eu/dailydata/index.php. Accessed, November 20, 2025.

[11] N. Szabo, *The Idea of Smart Contracts*, 1997. https://nakamotoinstitute.org/library/the-idea-of-smart-contracts/. Accessed, November 13, 2025.

[12] V. Buterin, *A Next-Generation Smart Contract and Decentralized Application Platform*, Ethereum Whitepaper, 2014. https://ethereum.org/whitepaper/. Accessed, November 6, 2025.

[13] G. Wood, *Ethereum: A Secure Decentralised Generalised Transaction Ledger*, Ethereum Yellow Paper, 2014. https://cryptodeep.ru/doc/paper.pdf. Accessed, November 8, 2025.

[14] A. M. Antonopoulos and G. Wood, *Mastering Ethereum*, O'Reilly Media, 2018. https://github.com/ethereumbook/ethereumbook. Accessed, November 8, 2025.