

Time Series Analysis of Confidentiality Degradation in Encrypted Search Systems

Alexander Towell *
Southern Illinois University-Edwardsville Math Department

April 30, 2021

Abstract

We present a time series analysis of confidentiality degradation in encrypted search systems subject to known-plaintext attacks. By modeling the adversary’s accuracy as a time-dependent confidentiality measure, we develop forecasting models based on ARIMA and dynamic regression techniques. Our analysis provides quantitative estimates for how quickly adversary knowledge accumulates, enabling organizations to establish data-driven policies for password rotation and system reinitialization that maintain specified confidentiality thresholds.

Contents

1	Introduction	2
2	Related work	3
3	Encrypted search model	3
4	Threat model: known-plaintext attack	4
5	Confidentiality measure	5
5.1	Security threshold and password rotation	5
5.2	Forecasting model	6
6	Data description	6
6.1	Data generation	6

*atowell@siue.edu

7	Time series analysis of $\{\pi_t\}$	7
7.1	Visualization and stationary transformations	7
7.2	ARIMA model selection	9
7.3	Model construction and evaluation	10
7.3.1	IMA(1,1)	10
7.3.2	IMA(1,2)	11
7.3.3	ARIMA(1,1,2)	11
7.4	Forecasting	13
7.5	Incorporating <i>a priori</i> information	14
8	Future work: incorporating covariates	17
8.1	Information-theoretic covariates	17
8.2	Side-channel information	18
9	Conclusion	18
9.1	Model comparison	18
9.2	Security policy implications	18
9.3	Limitations and future directions	19

1 Introduction

Cloud computing enables organizations to store and process data on remote infrastructure, but this convenience comes at the cost of reduced control over data confidentiality. Even when data is encrypted at rest, system administrators and other insiders pose a significant threat. This tension between utility and security has motivated the development of encrypted search systems.

Encrypted search attempts to resolve the fundamental trade-off between confidentiality and usability by enabling authorized users to query encrypted data without revealing query contents to untrusted parties.

Definition 1. *Encrypted search allows authorized search agents to investigate the presence of specific search terms in a confidential target data set, such as a database of encrypted documents, while the contents—especially the meaning of the target data set and search terms—are hidden from unauthorized personnel, including the system administrators of a cloud server.*

Essentially, encrypted search enables oblivious queries where users can search confidential databases on untrusted systems without revealing their information needs (and in more sophisticated systems, without revealing which documents matched the query).

We define the *adversary* as any untrusted party with access to the remote system where confidential data is stored.¹

¹System administrators are typical examples.

However, perfect confidentiality is not achievable in practice. Encrypted search systems leak information through access patterns, query frequencies, and other side channels. In this paper, we focus on an adversary who attempts to infer the plaintext queries by analyzing the history of encrypted searches using a known-plaintext attack.

We quantify confidentiality using the proportion of queries the adversary successfully decodes. Since confidentiality degrades as the adversary observes more queries, we model this degradation as a time series process.

Time series analysis enables us to forecast future confidentiality levels, answering critical security questions such as: How long until confidentiality falls below an acceptable threshold? When should passwords be reset to maintain a minimum confidentiality level?

These forecasts support data-driven security policies. Password resets impose usability and security costs, but delaying them risks unacceptable confidentiality loss. Our analysis provides quantitative guidance for balancing these competing concerns.

2 Related work

Searchable encryption was introduced by Song, Wagner, and Perrig [5], who developed practical techniques for searching on encrypted data while preserving query confidentiality. Their foundational work demonstrated that encrypted search systems could achieve practical performance, but acknowledged that such systems necessarily leak some information through access patterns and query frequencies.

Subsequent research has shown that these information leakages enable significant attacks. Islam et al. [3] demonstrated access pattern disclosure attacks that exploit the deterministic nature of encrypted search schemes. By observing which encrypted documents match encrypted queries, an adversary can infer plaintext queries even without directly observing query contents. Their attack model closely parallels our known-plaintext attack framework: both exploit frequency analysis and co-occurrence patterns in the query stream to recover plaintext information. While Islam et al. focused on one-time attacks, our work extends this perspective by modeling confidentiality degradation as a time series process, enabling forecasts of when adversary knowledge will violate security thresholds.

3 Encrypted search model

In information retrieval, a *search agent* submits a *query* representing an *information need* to an information system, which responds with relevant objects (e.g., documents).

We make the following simplifying assumptions about the query model and encryption scheme.

Assumption 1. *Queries consist of sequences of keywords (a bag-of-words model).*

Assumption 2. *The encrypted search system uses a deterministic encryption scheme where each plaintext keyword maps to a unique encrypted token (trapdoor).*

More formally, we define the encryption function as

$$h: \mathbb{X} \mapsto \mathbb{Y}, \tag{1}$$

where \mathbb{X} is the set of plaintext search keywords and \mathbb{Y} is the set of encrypted tokens (trapdoors).

Definition 2 (Trapdoor). *A trapdoor is a cryptographic token generated by applying h to a plaintext keyword, enabling the untrusted system to perform searches without learning the plaintext.*

Since h is injective, there exists an inverse function

$$g: \mathbb{Y} \mapsto \mathbb{X} \quad (2)$$

such that $x = g(h(x))$ for every $x \in \mathbb{X}$.

Definition 3. *The adversary can observe the sequence of encrypted queries (trapdoors) submitted by authorized search agents but does not initially know g .*

The security objective is to prevent the adversary from learning g and thus decoding the queries.

We now formalize the time series notation. Following standard conventions, we use uppercase letters for random variables and lowercase for realizations. A time series $\{Y_t\}$ denotes a sequence of random variables indexed by time t , with $\{y_t\}$ denoting an observed realization.

The plaintext queries form a discrete-time, discrete-valued time series $\{x_t\}$ where $x_t \in \mathbb{X}$ is the t -th keyword submitted by search agents. The adversary cannot observe $\{x_t\}$ directly.

Definition 4. *The encrypted query sequence $\{c_t\}$ is defined by*

$$c_t = h(x_t),$$

where $c_t \in \mathbb{Y}$ is the trapdoor corresponding to plaintext x_t .

We model queries probabilistically as a random process $\{X_t\}$ with distribution determined by a language model:

$$\Pr(X_t = x_t | X_{t-1} = x_{t-1}, \dots, X_1 = x_1). \quad (3)$$

This formulation assumes queries depend only on query history, not on external factors such as user identity or time of day. We discuss extensions to include such covariates in Section 8.

The encrypted query sequence is then modeled as the random process $\{C_t\}$ where $C_t = h(X_t)$.

4 Threat model: known-plaintext attack

The adversary observes the encrypted query sequence $\{c_t\}$ and attempts to recover the plaintext sequence $\{x_t\}$ by learning the decryption function g . We focus on frequency-based cryptanalysis, specifically known-plaintext attacks. Side-channel information is not considered in this work; we discuss potential extensions in Section 8.

Assumption 3. *The decryption function g is unknown to the adversary.*

The adversary's strategy is to estimate g by exploiting statistical properties of natural language. Given observed ciphers $\{c_1, \dots, c_T\}$, a maximum likelihood estimator of g is

$$\hat{g} = \arg \max_{g \in G} \Pr(X_1 = g(c_1)) \prod_{t=2}^T \Pr(X_t = g(c_t) | X_{t-1} = g(c_{t-1}), \dots, X_1 = g(c_1)), \quad (4)$$

where G denotes the set of all injective functions from \mathbb{Y} to \mathbb{X} .

Perfect confidentiality would require that all plaintexts be equally likely, making them indistinguishable to the adversary. In this case, \hat{g} would be inconsistent. However, natural language exhibits strong statistical regularities (word frequencies follow Zipf's law, co-occurrence patterns are non-uniform), enabling the adversary to learn something about $\{x_t\}$ from $\{c_t\}$.

In practice, the adversary does not know the true distribution of $\{X_t\}$ but can approximate it using external corpora.

Assumption 4. *The adversary has access to an approximate language model $\hat{\Pr}(\cdot)$ for the query distribution.*

Definition 5. *In a known-plaintext attack, the adversary estimates g by solving the MLE in Equation (4) with the true distribution replaced by the approximate distribution $\hat{\Pr}(\cdot)$.*

The quality of the attack depends on how well $\hat{\Pr}(\cdot)$ approximates the true query distribution.

5 Confidentiality measure

We quantify confidentiality degradation using the adversary's decryption accuracy.

Definition 6. *The confidentiality measure $\{\pi_t\}$ is defined as the fraction of ciphers correctly decoded by the adversary using \hat{g} :*

$$\pi_t = \frac{\delta_t}{Nt}, \quad (5)$$

where

$$\delta_t = \sum_{t'=1}^{Nt} \mathbf{1}[g(c_{t'}) = \hat{g}(c_{t'})]. \quad (6)$$

Here N is a sampling parameter: we compute one measurement of π_t after every N queries.

The measure π_t represents the proportion of the entire query history that the adversary can decode at time t . Note that π_t increases as the adversary observes more queries and refines \hat{g} .

5.1 Security threshold and password rotation

Suppose we require that confidentiality remain above a threshold $\pi^* \in (0, 1)$. We define the critical time T^* as

$$T^* = \min\{T : \pi_T > \pi^*\}. \quad (7)$$

At time T^* , the adversary can decode more than proportion π^* of queries, violating the security requirement.

To maintain the threshold, we must reinitialize the encryption before reaching T^* . This typically involves changing the cryptographic key (e.g., requiring password changes), which generates a new encryption function h' and resets the adversary's knowledge. The adversary must then restart the attack process with \hat{g}' .

Forecasting $\{\pi_t\}$ enables proactive security policies: we can predict when π_t will exceed π^* and schedule reinitialization accordingly.

5.2 Forecasting model

Since $\{c_t\}$ is random, the confidentiality measure $\{\pi_t\}$ is also random. We model it as a stochastic process $\{\Pi_t\}$.

Given observations π_1, \dots, π_T , the h -step ahead forecast distribution is

$$\Pi_{T+h|T} \equiv (\Pi_{T+h} \mid \Pi_1 = \pi_1, \dots, \Pi_T = \pi_T).$$

The forecast mean is denoted $\mathbb{E}[\Pi_{T+h|T}]$.

Our goal is to estimate the forecast distribution from observed data. In practice, we estimate the forecast mean $\hat{\pi}_{T+h|T}$ and construct prediction intervals using the estimated forecast variance.

6 Data description

We analyze simulated data to evaluate forecasting models for $\{\pi_t\}$. The simulation generates realistic query sequences and models an adversary conducting a known-plaintext attack.

6.1 Data generation

The simulation proceeds in two phases: query generation and attack simulation.

6.1.0.1 Query generation. We first generate a synthetic plaintext query sequence $\{x_t\}$:

1. A bigram language model is estimated from a text corpus. (The specific corpus has been lost; it was a general English text collection.)
2. Plaintext queries $\{x_t\}$ are sampled from this bigram model.
3. Each plaintext x_t is encrypted via a cryptographic hash function to produce ciphers $c_t = h(x_t)$.

6.1.0.2 Attack simulation. Given the cipher sequence $\{c_t\}$, we simulate an adversary attempting to learn g :

1. After every $N = 50$ ciphers, the adversary updates its estimate \hat{g} by solving a unigram MLE:

$$\hat{g}_T = \arg \max_{g \in G} \prod_{t=1}^T \hat{\Pr}(X_t = g(c_t)).$$

Note that the adversary uses a unigram model even though queries were generated from a bigram model. This reflects information loss in practice.

2. The adversary's language model $\hat{\Pr}(\cdot)$ is estimated from a different corpus than the one used to generate $\{x_t\}$. This mismatch makes \hat{g}_T inconsistent (it does not converge to g as $T \rightarrow \infty$), reflecting realistic conditions where the adversary's knowledge is imperfect.
3. The confidentiality measure π_t is computed using the current estimate \hat{g}_t .

This setup models a realistic scenario where the adversary has approximate but not perfect knowledge of the query distribution.

7 Time series analysis of $\{\pi_t\}$

The adversary’s accuracy exhibits temporal dependence: accuracy at time t is correlated with recent values $\pi_{t-1}, \pi_{t-2}, \dots$, with correlation typically decreasing as lag increases. This autocorrelation structure motivates time series methods for forecasting.

We partition the data into training and test sets. The training set is used for model fitting and the test set is held out for evaluation. Sample training data:

```
## [1] 0.358159 0.351208 0.347271 0.346403 0.352666 0.350445
```

7.1 Visualization and stationary transformations

ARIMA models assume stationarity: constant mean, constant variance, and autocorrelation that depends only on lag. We must transform $\{\pi_t\}$ to approximate stationarity before fitting ARIMA models.

Figure 1 plots the training data. The series exhibits an upward trend, indicating non-stationarity in the mean.

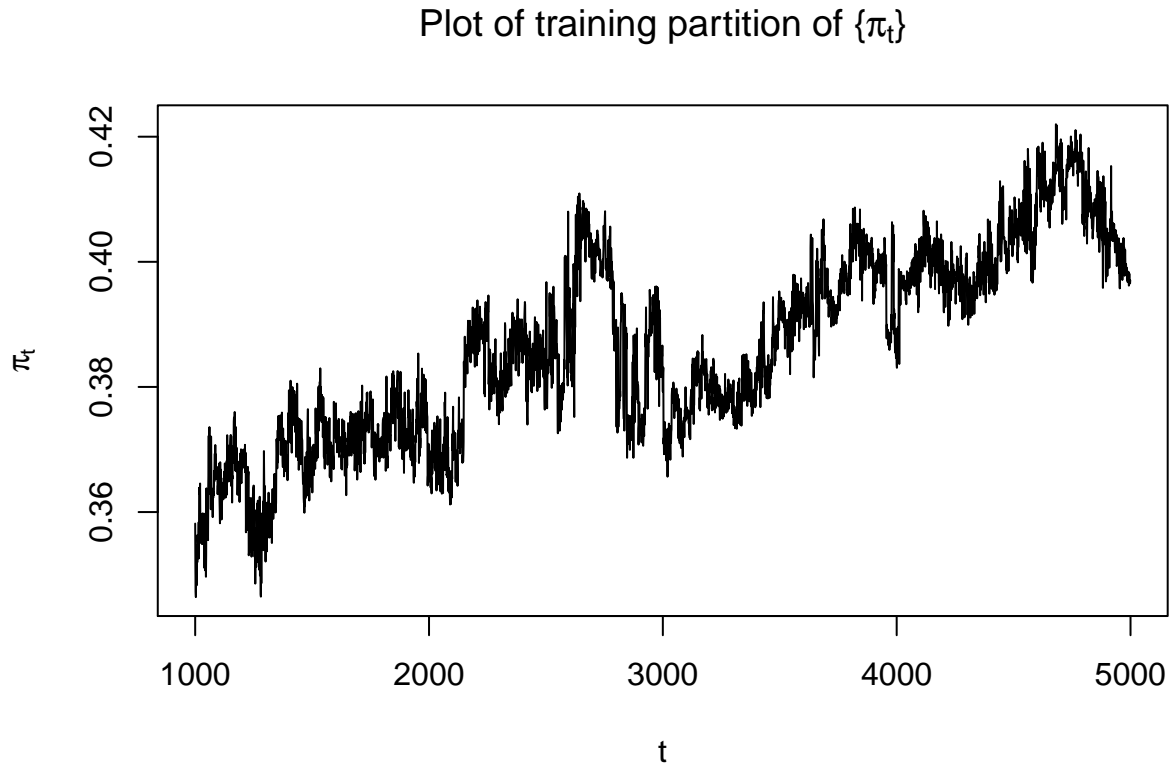


Figure 1: Time series plot of $\{\pi_t\}$ showing non-stationary behavior.

Figure 2 shows the sample ACF and PACF. The slow decay in the ACF confirms non-stationarity.

The variance appears reasonably constant, so variance-stabilizing transformations (e.g., logarithm) are unnecessary. To remove the trend, we apply differencing: the d -th order differenced series is

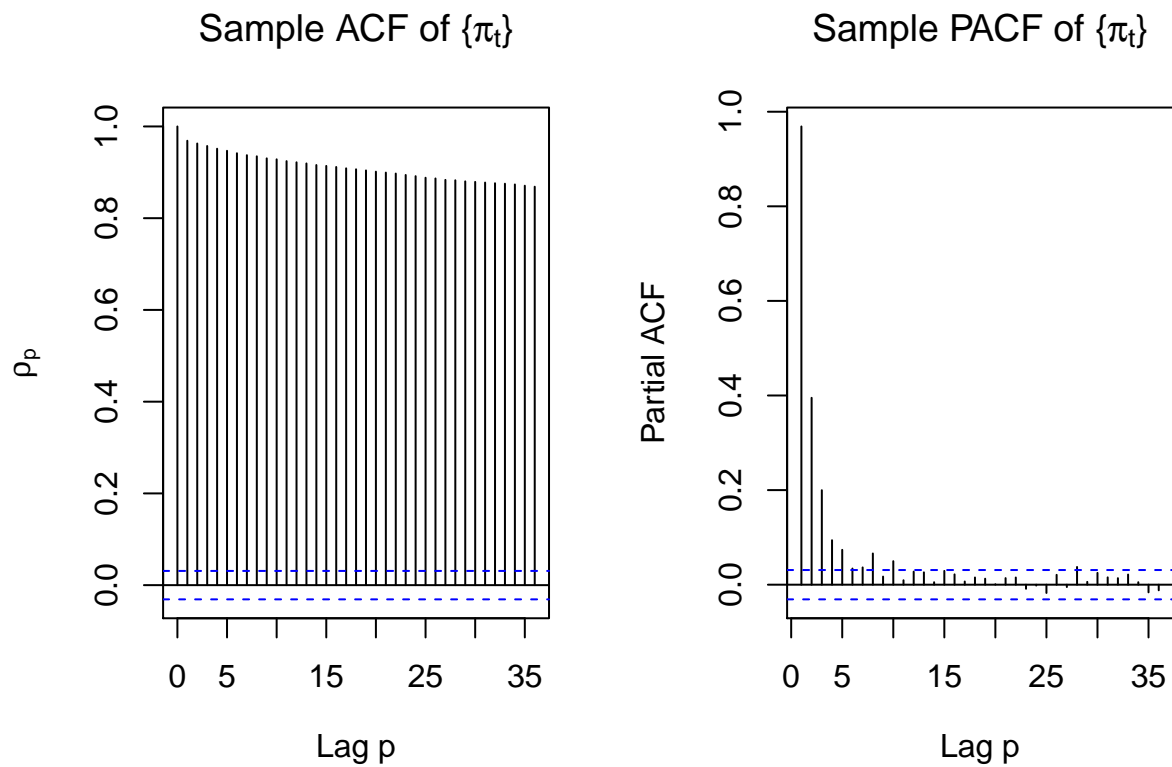


Figure 2: Sample ACF and PACF showing strong autocorrelation.

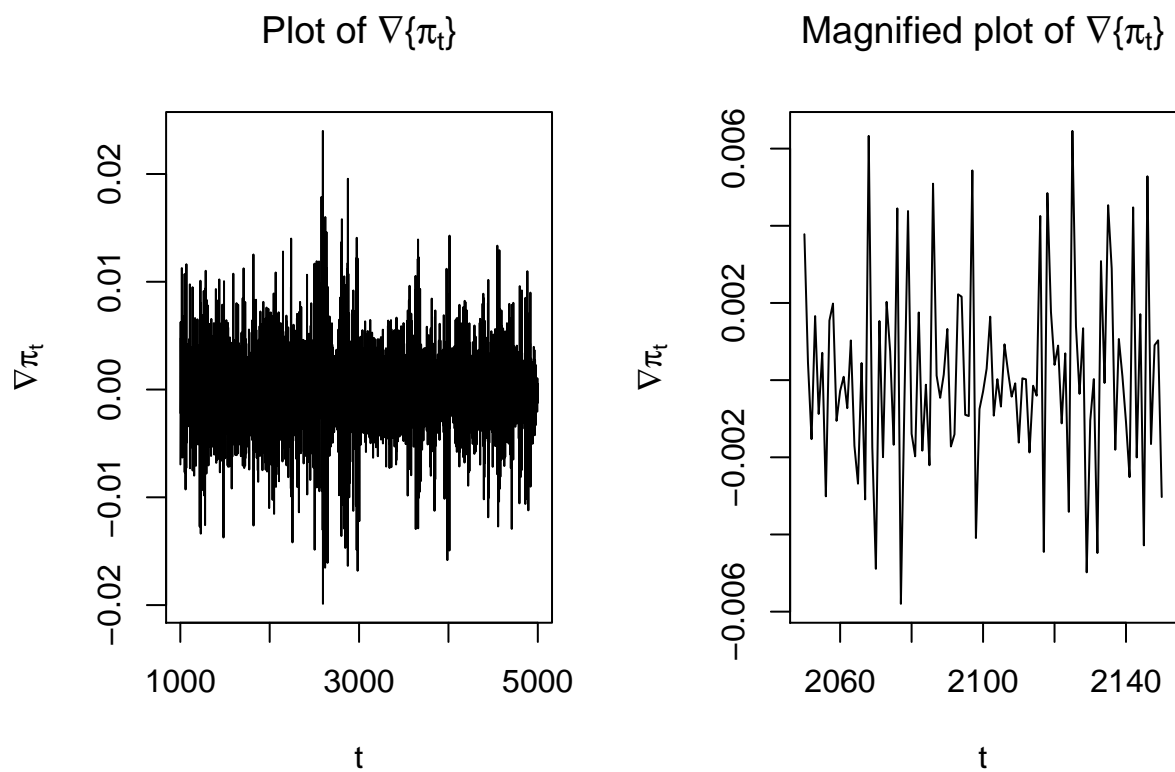


Figure 3: First-differenced series $\nabla\{\pi_t\}$.

denoted $\nabla^d\{\pi_t\}$. Differencing is a flexible, non-parametric approach that adapts to complex trend patterns without assuming a specific functional form.

Figure 3 shows the differenced series $\nabla\{\pi_t\}$. The trend is removed, values fluctuate around zero, and variance appears constant. This suggests stationarity.

We apply the augmented Dickey-Fuller test [2] to formally test for stationarity:

```
## Augmented Dickey-Fuller Test
## Dickey-Fuller = -20.37, Lag order = 15, p-value = 0.01
## alternative hypothesis: stationary
```

The p -value of 0.01 provides strong evidence against the null hypothesis of non-stationarity. We conclude that $\nabla\{\pi_t\}$ is approximately stationary and proceed with ARIMA modeling.

7.2 ARIMA model selection

Our goal is to build a parsimonious forecasting model. Following Occam's razor, simpler models with fewer parameters are preferred: they generalize better to unseen data and avoid overfitting [1]. However, the model must be sufficiently complex to capture the data's structure. This is the classical bias-variance tradeoff.

ARIMA(p, d, q) models are parameterized by three integers: p (autoregressive order), d (differencing order), and q (moving average order). We seek small values of p , d , and q that adequately fit the data. While information criteria like AIC can guide selection, we also examine diagnostic plots.

Figure 4 shows the ACF and PACF of $\nabla\{\pi_t\}$.

The ACF cuts off after lag 1, and the PACF decays exponentially, suggesting an MA(1) process for $\nabla\{\pi_t\}$ (i.e., IMA(1,1) for $\{\pi_t\}$).

The extended ACF (EACF) table provides additional guidance:

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x o o o o o o x x x o o o o
## 1 x x o o o o x o o o o o o o
## 2 x x x o o o x o o o o o o o
## 3 x x x x o o o o o o o o o o
## 4 x x x x x o o o o o o o o o
## 5 x x x x x x o o o o o o o o
## 6 x x x x x o x o o o o o o o
## 7 x x x x x x x o o o o o o o
```

The EACF suggests candidate models ARMA(0, 1), ARMA(0, 2), and ARMA(1, 2) for $\nabla\{\pi_t\}$. We evaluate all three.

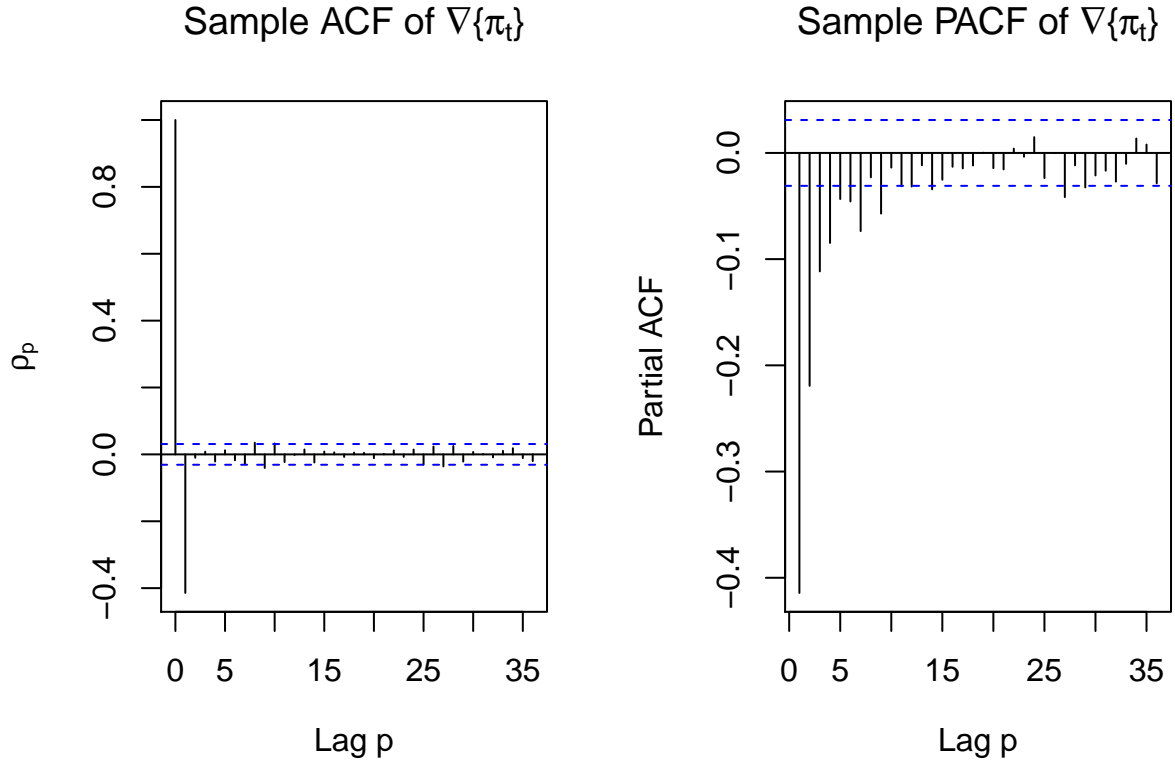


Figure 4: Sample ACF and PACF of the differenced series $\nabla\{\pi_t\}$.

7.3 Model construction and evaluation

Model adequacy is assessed through residual diagnostics. A well-specified model produces residuals $\{e_t\}$ that behave like white noise:

1. Uncorrelated: If residuals are correlated, the model has not captured all temporal structure and can be improved.
2. Zero mean: Non-zero mean residuals indicate systematic bias in forecasts.
3. Constant variance: Heteroscedasticity may require variance modeling.

7.3.1 IMA(1,1)

We first fit $\text{ARIMA}(0, 1, 1)$, the simplest candidate model. The estimated parameter is:

```
##      ma1
## -0.5705635
```

Figure 5 shows residual diagnostics.

The histogram appears symmetric around zero, but the Q-Q plot shows deviations from normality. More critically, the ACF and PACF exhibit significant residual autocorrelation, indicating the model has not captured all temporal dependence. We reject this model.

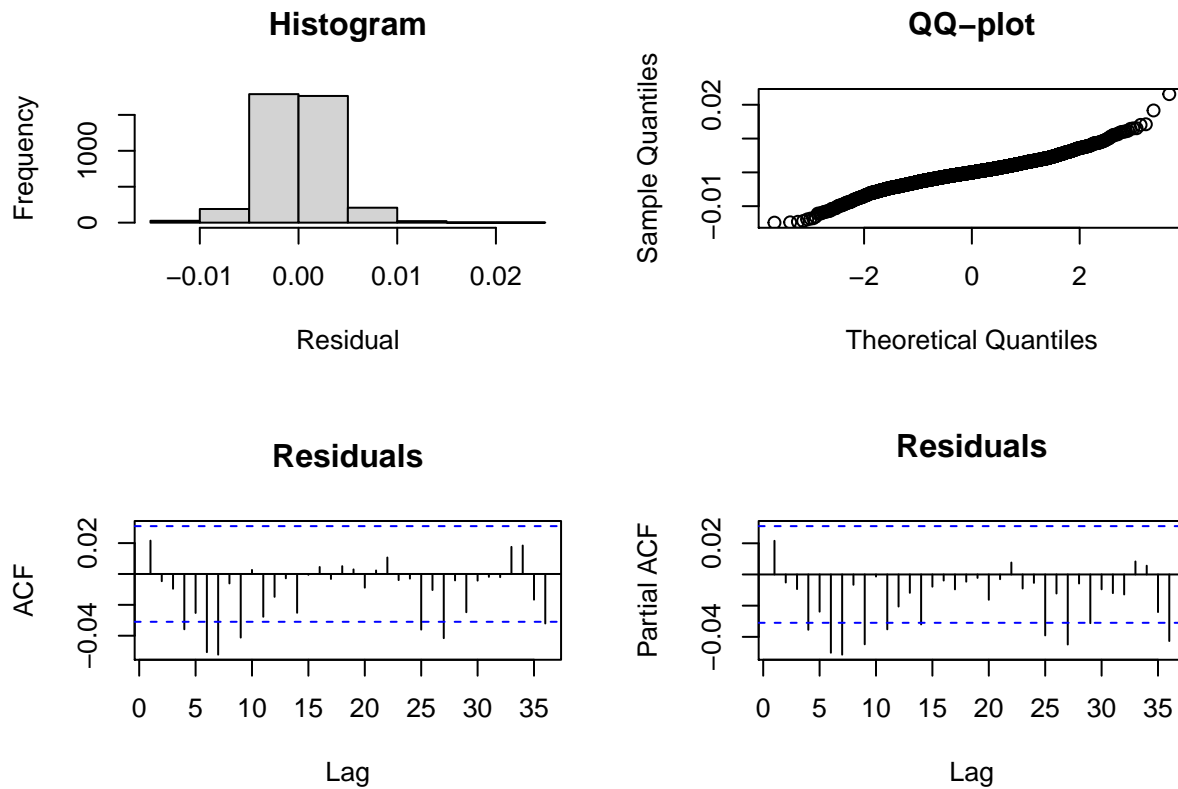


Figure 5: Residual diagnostics for ARIMA(0, 1, 1).

7.3.2 IMA(1,2)

Next we fit ARIMA(0, 1, 2). The estimated parameters are:

```
##          ma1          ma2
## -0.55254743 -0.04124893
```

Figure 6 shows residual diagnostics.

As before, the histogram is symmetric, and the Q-Q plot indicates non-normality (though normality is not strictly required for forecasting). However, the ACF and PACF again show significant residual correlation. This model is also inadequate. We reject it.

7.3.3 ARIMA(1,1,2)

Finally, we fit ARIMA(1, 1, 2). The estimated parameters are:

```
##          ar1          ma1          ma2
##  0.8839643 -1.4423574  0.4689439
```

Figure 7 shows residual diagnostics.

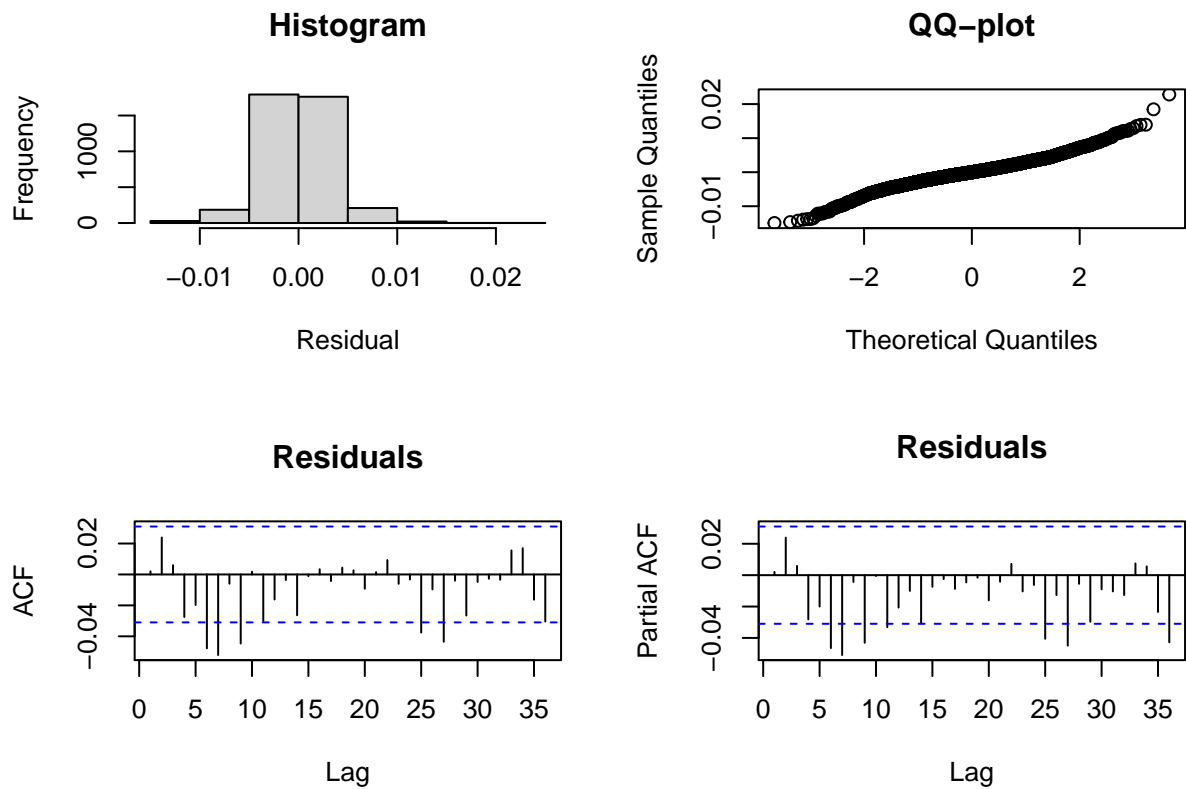


Figure 6: Residual diagnostics for ARIMA(0, 1, 2).

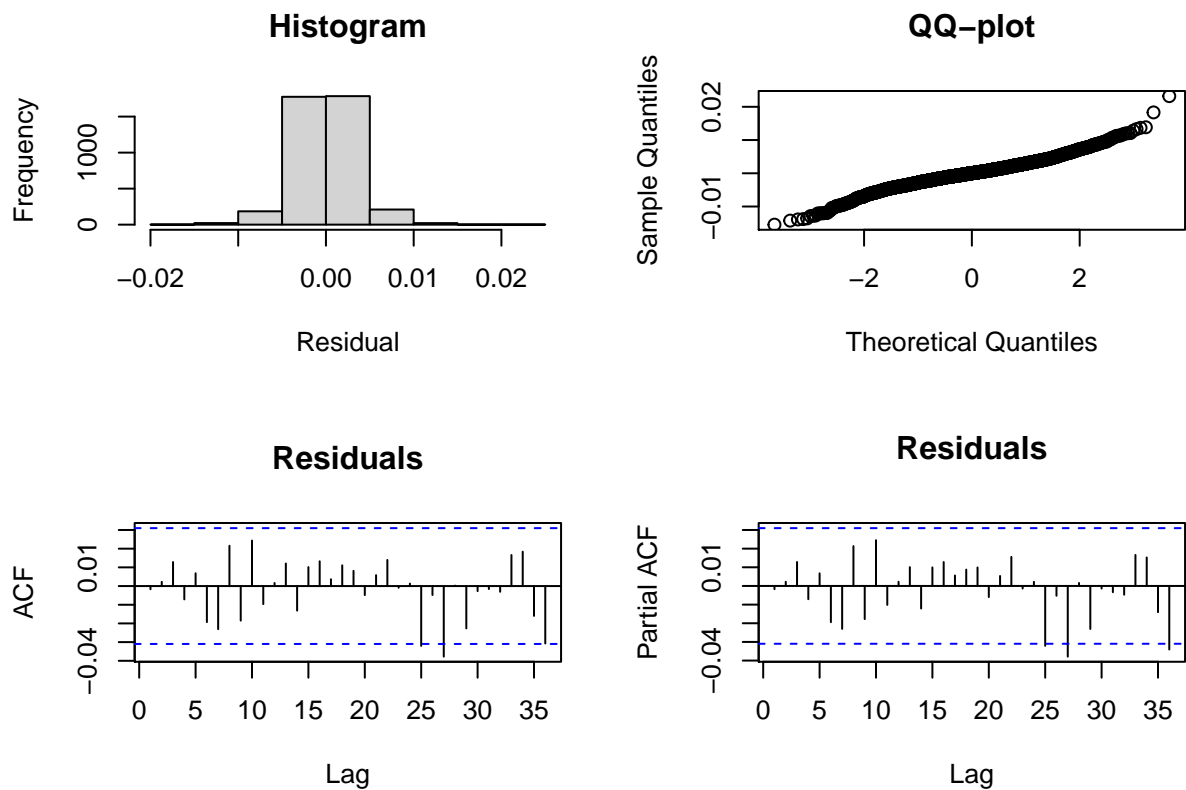


Figure 7: Residual diagnostics for ARIMA(1, 1, 2).

Again, the histogram is symmetric and the Q-Q plot shows non-normality. However, the ACF and PACF appear acceptable: no significant residual autocorrelation is evident.

We apply the Ljung-Box test (lag 10) to formally assess the white noise hypothesis:

```
## Box-Ljung test
## data: model.3$residuals
## X-squared = 10.333, df = 7, p-value = 0.1705
```

The null hypothesis is that residuals are white noise. The p -value of 0.171 provides substantial support for this hypothesis.

We select ARIMA(1,1,2) as our model. Since this is the only model passing diagnostics, formal comparison via AIC is unnecessary.

The fitted model summary is:

```
## ARIMA(1,1,2)
##
## Coefficients:
##          ar1          ma1          ma2
##          0.8840   -1.4424    0.4689
## s.e.    0.0276    0.0338    0.0266
##
## sigma^2 estimated as 1.088e-05:  log likelihood=17177.64
## AIC=-34347.27  AICc=-34347.26  BIC=-34322.1
```

In operator form, the model is

$$(1 - 0.884 B)\nabla\Pi_t = (1 + 1.442 B - 0.469 B^2)e_t, \quad (8)$$

which factors as

$$(1 - 0.884 B)\nabla\Pi_t = (1 - 0.273 B)(1 + 1.715 B)e_t, \quad (9)$$

where $e_t \sim \text{WN}(0, \sigma^2)$ with $\sigma = 0.0033$.

7.4 Forecasting

We now use the fitted ARIMA(1,1,2) model to forecast future confidentiality degradation. At time $T = 5000$, we compute forecasts $\hat{\pi}_{T+h|T}$ for $h = 1, \dots, 1000$.

Figure 8 shows the forecast along with 80% prediction intervals.

The held-out test data falls within the 80% prediction interval for most time points, suggesting reasonable forecast performance. However, the prediction intervals are quite wide. We hypothesize this reflects model misspecification: the purely autoregressive structure may not capture the underlying process adequately. Theoretically, π_t should approach an asymptotic limit as the adversary's knowledge saturates, but the ARIMA model does not incorporate this constraint. We explore theory-driven models in the next section.

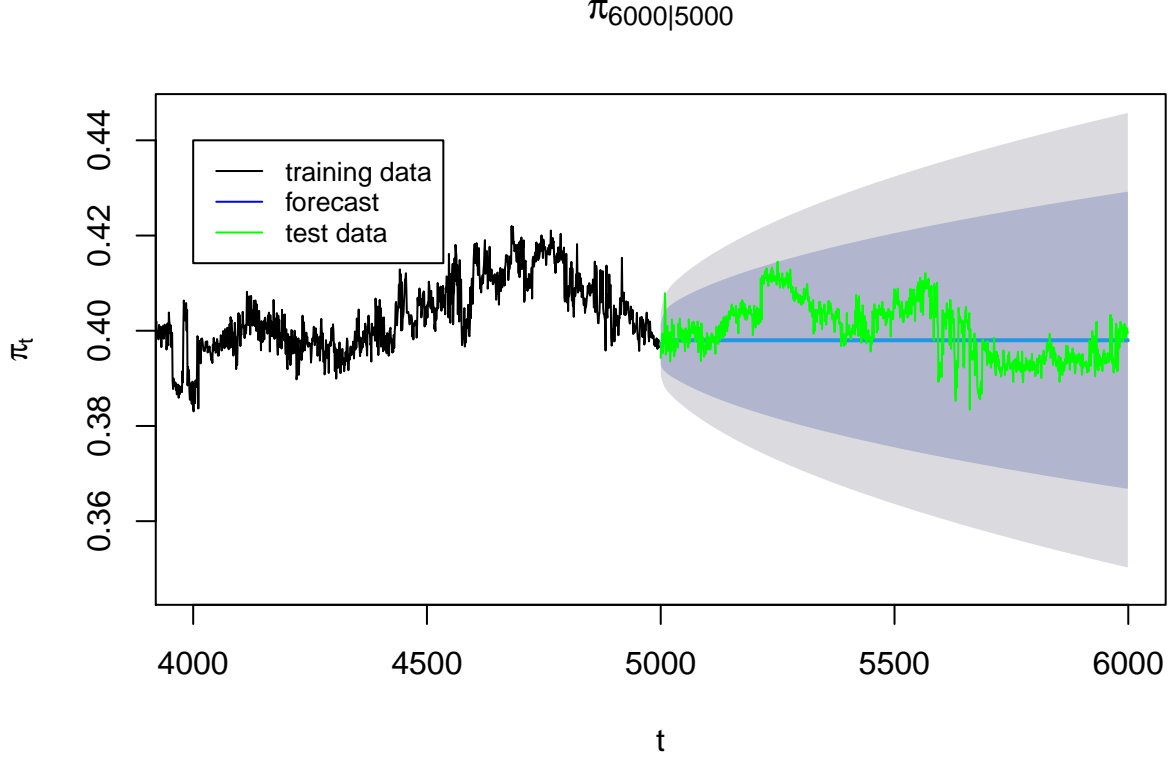


Figure 8: ARIMA(1, 1, 2) forecasts with training and test data.

7.5 Incorporating *a priori* information

We now incorporate domain knowledge about the confidentiality measure. Assuming the plaintext distribution and encryption function remain fixed, we expect:

1. $\pi_t \in [0, 1]$ (it is a proportion).
2. π_t should be monotonically increasing: observing more queries cannot decrease the adversary's accumulated knowledge.
3. π_t should approach an asymptotic limit $c \leq 1$ as the adversary's knowledge saturates.

These properties suggest models like Gompertz growth curves or scaled CDFs. For simplicity, we approximate the trend with a logarithmic function:

$$\pi_t = \beta_0 + \beta_1 \log t. \quad (10)$$

While this lacks an asymptote, it provides a reasonable approximation over finite time horizons.

We model deviations from this trend as ARIMA errors:

$$\Pi_t = \beta_0 + \beta_1 \log t + \eta_t, \quad (11)$$

where $\eta_t \sim \text{ARIMA}(p, d, q)$.

Note that this formulation (regression with ARIMA errors) differs from standard regression. As Hyndman [4] notes, the regression coefficients condition on lagged responses, complicating interpretation. Nevertheless, this model can improve forecasts by capturing both the deterministic trend and stochastic dynamics.

We fit this model to the data:

```
## Regression with ARIMA(1,1,2) errors
##
## Coefficients:
##          ar1          ma1          ma2          xreg
##          0.8521    -1.4012    0.4355    0.0292
## s.e.      0.0234     0.0277    0.0211    0.0202
##
## sigma^2 estimated as 6.828e-06:  log likelihood=45279.87
## AIC=-90549.73  AICc=-90549.73  BIC=-90513.68
```

This uses ARIMA(1,1,2) errors with a logarithmic trend term. (We also tested ARIMA(2,1,2), which yielded tighter prediction intervals but was not substantially better.)

Figure 9 shows residual diagnostics.

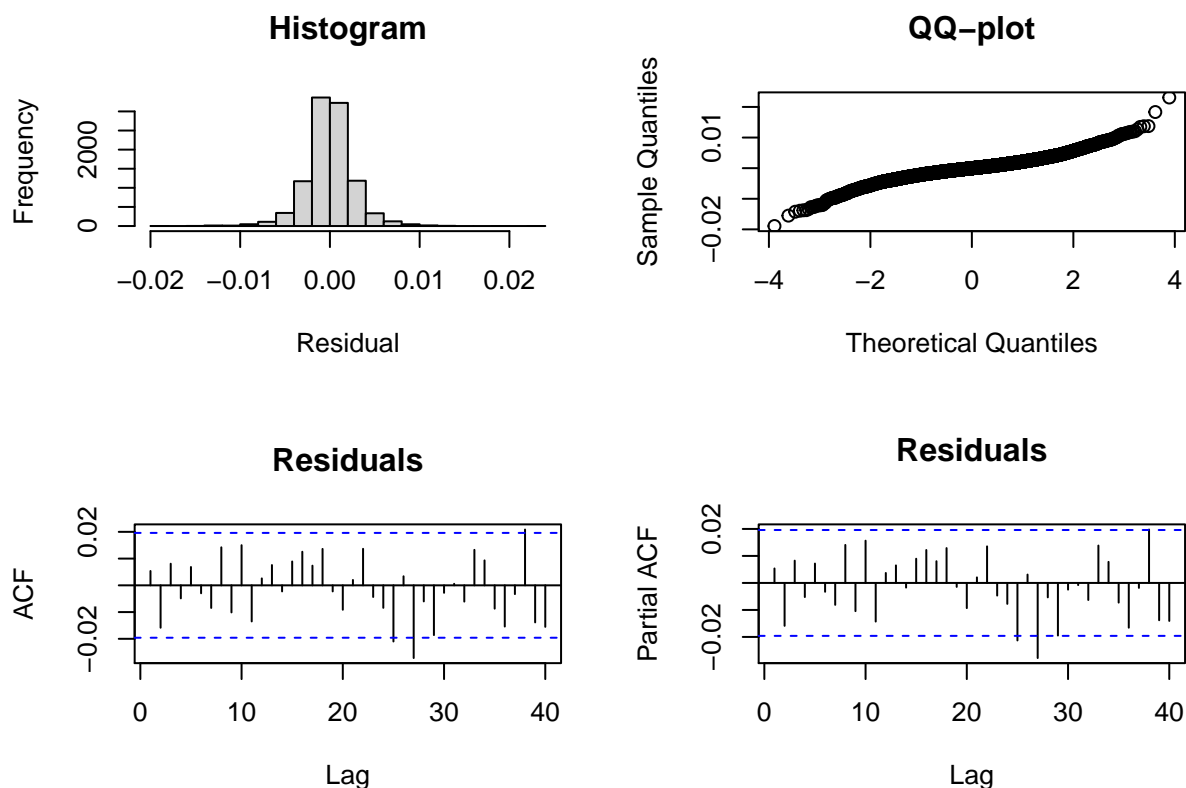


Figure 9: Residual diagnostics for regression with ARIMA(1,1,2) errors.

The histogram is symmetric, the Q-Q plot shows non-normality, and the ACF and PACF appear acceptable. The Ljung-Box test (lag 10):

```
## Box-Ljung test
## data:  reg_model$residuals
## X-squared = 10.279, df = 6, p-value = 0.1134
```

At the 5% significance level, the model is marginally adequate ($p = 0.113$). The fitted model is

$$\hat{\Pi}_t = 0.029 \log t + \eta_t, \quad (12)$$

where $\eta_t \sim \text{ARIMA}(1, 1, 2)$ with coefficients as above and $e_t \sim \text{WN}(0, \sigma^2)$ with $\sigma = 0.0026$.

Figure 10 shows forecasts from this model along with training and test data.

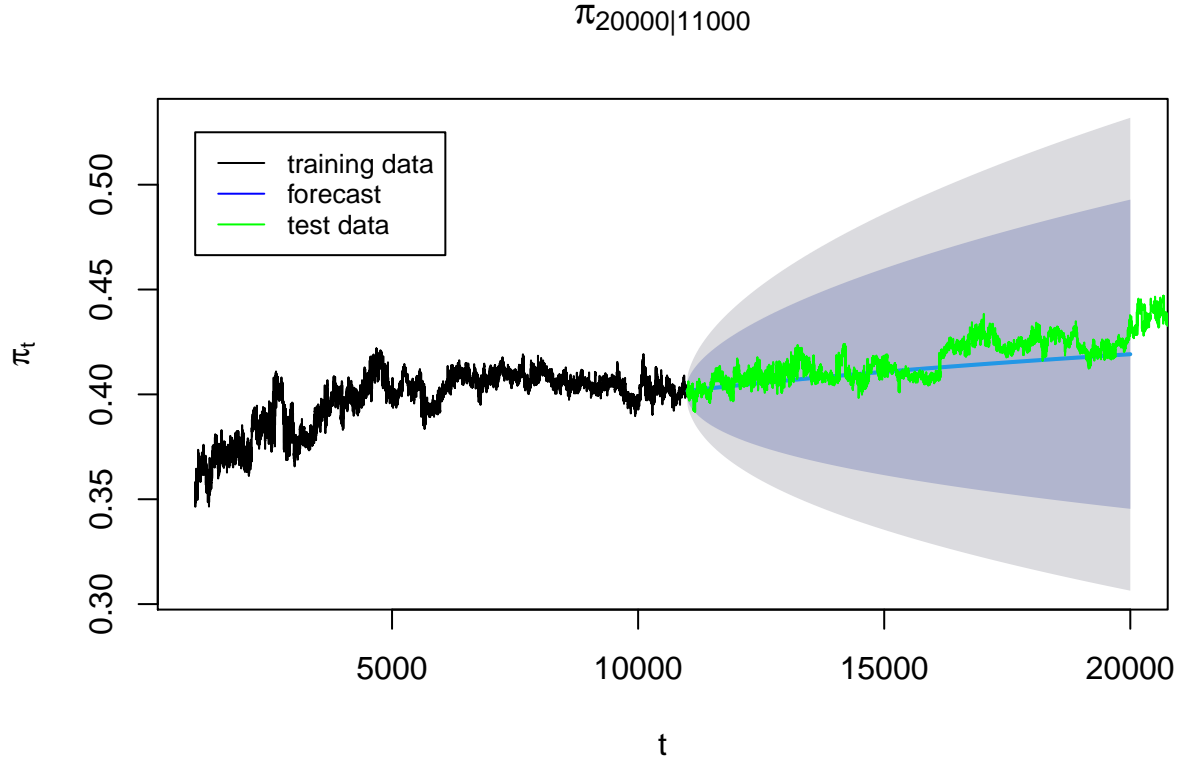


Figure 10: Forecasts from regression with ARIMA errors model.

The forecasts extend much further into the future and follow the gradual upward trend more faithfully than the pure ARIMA model.

For comparison, we also fit an $\text{ARIMA}(1, 1, 2)$ model with drift:

```
##          ar1          ma1          ma2          drift
## 8.472755e-01 -1.396547e+00 4.324823e-01 4.695287e-06
```

The drift coefficient is small but positive. Over long horizons, this accumulates to produce an upward trend (Figure 11).

However, ARIMA with drift implies linear growth, which is theoretically inappropriate: π_t must remain bounded in $[0, 1]$. The logarithmic regression model better respects this constraint.

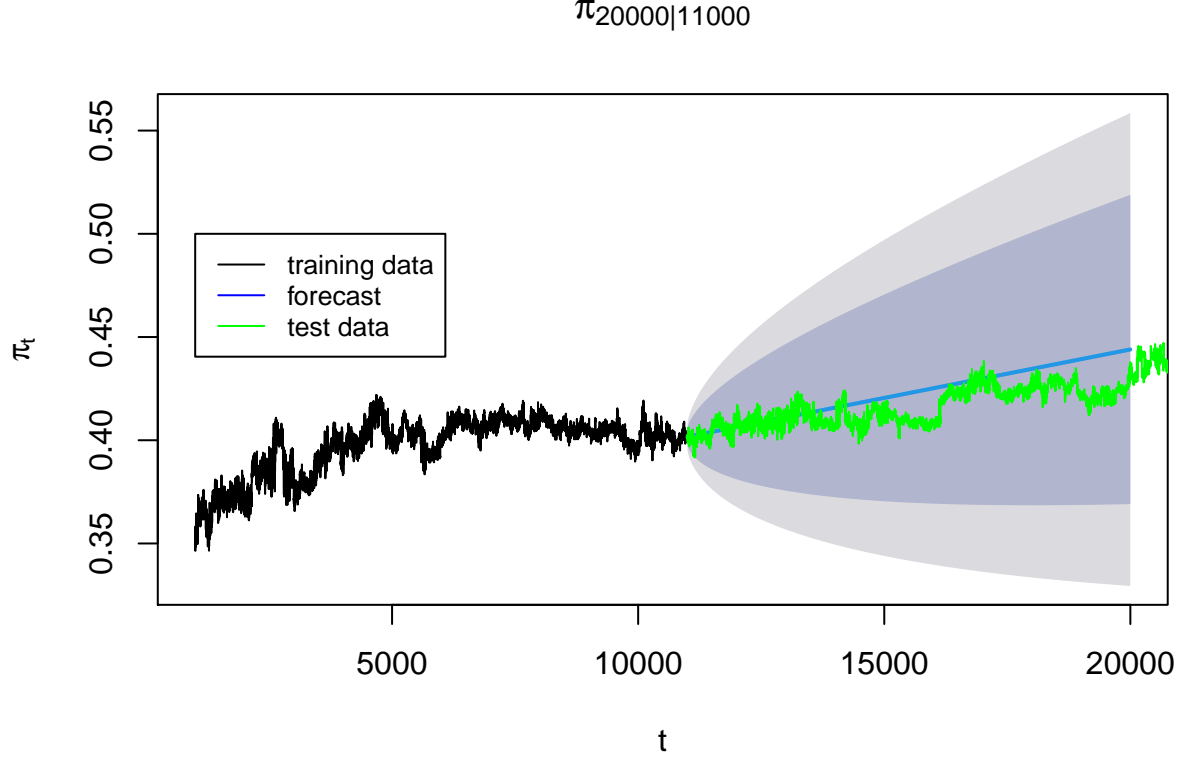


Figure 11: Forecasts from ARIMA(1, 1, 2) with drift.

8 Future work: incorporating covariates

Our forecasting models use only lagged values of π_t to predict future confidentiality. This approach captures temporal patterns but ignores potential covariates that may affect adversary performance. Incorporating such covariates could improve forecast accuracy and provide insights into factors driving confidentiality degradation.

8.1 Information-theoretic covariates

The adversary learns from observing ciphers $\{c_t\}$, but not all observations are equally informative. We could model the information content of each observation explicitly.

Define the information content of observation t as

$$I_t = -\log_2 \Pr(g(c_t)), \quad (13)$$

measured in bits. Observations with low probability under the adversary’s model carry more information.

Lagged values of I_t could be used as predictors in a dynamic regression:

$$\Pi_t = \beta_0 + \beta_1 I_{t-1} + \beta_2 I_{t-2} + \dots + \eta_t. \quad (14)$$

Alternatively, cumulative information (empirical entropy) could serve as a covariate. This would allow the model to adapt to the informativeness of recent queries, potentially tightening prediction intervals during periods of rapid learning.

8.2 Side-channel information

Beyond the cipher sequence, the adversary may obtain side-channel information. For example:

- Timing information revealing query complexity
- Access patterns indicating document relevance
- External knowledge narrowing the plaintext space

Such events could be modeled as intervention variables or structural breaks. For instance, if the adversary learns that cipher c' maps to a subset $\mathbb{W} \subset \mathbb{X}$, this constitutes an information gain equivalent to reducing the entropy of the plaintext distribution. Dynamic regression could incorporate indicator variables for such events, allowing the model to capture sudden jumps in adversary accuracy.

9 Conclusion

We have developed time series models to forecast confidentiality degradation in encrypted search systems under known-plaintext attacks. Our analysis demonstrates that standard ARIMA models can capture the temporal dynamics of adversary accuracy, while dynamic regression with logarithmic trend terms better incorporates domain knowledge about the theoretical behavior of learning curves.

9.1 Model comparison

The pure ARIMA(1,1,2) model provides reasonable short-term forecasts and has the virtue of simplicity. However, it lacks theoretical grounding: ARIMA with drift predicts unbounded growth, eventually yielding impossible values ($\pi_t > 1$).

The logarithmic regression model ($\pi_t = \beta_0 + \beta_1 \log t + \eta_t$) better reflects the expected learning dynamics. While it also lacks an asymptote (eventually predicting $\pi_t > 1$ after approximately 1.5×10^{15} observations), this occurs far beyond any practical time horizon. The logarithmic form approximates the saturation behavior of adversary learning reasonably well over finite horizons.

9.2 Security policy implications

Consider a security threshold $\pi^* = 0.44$. Using our models, we estimate the critical time T^* at which confidentiality falls below this threshold. Figure 12 shows \hat{T}^* as a function of π^* .

For $\pi^* = 0.44$, we estimate $\hat{T}^* \approx 20000$, suggesting password reset should occur before 20,000 queries to maintain the threshold.

However, the 80% prediction intervals are substantially more conservative, sometimes recommending near-immediate password reset. This conservatism limits practical utility: overly frequent resets impose costs on users and administrators.

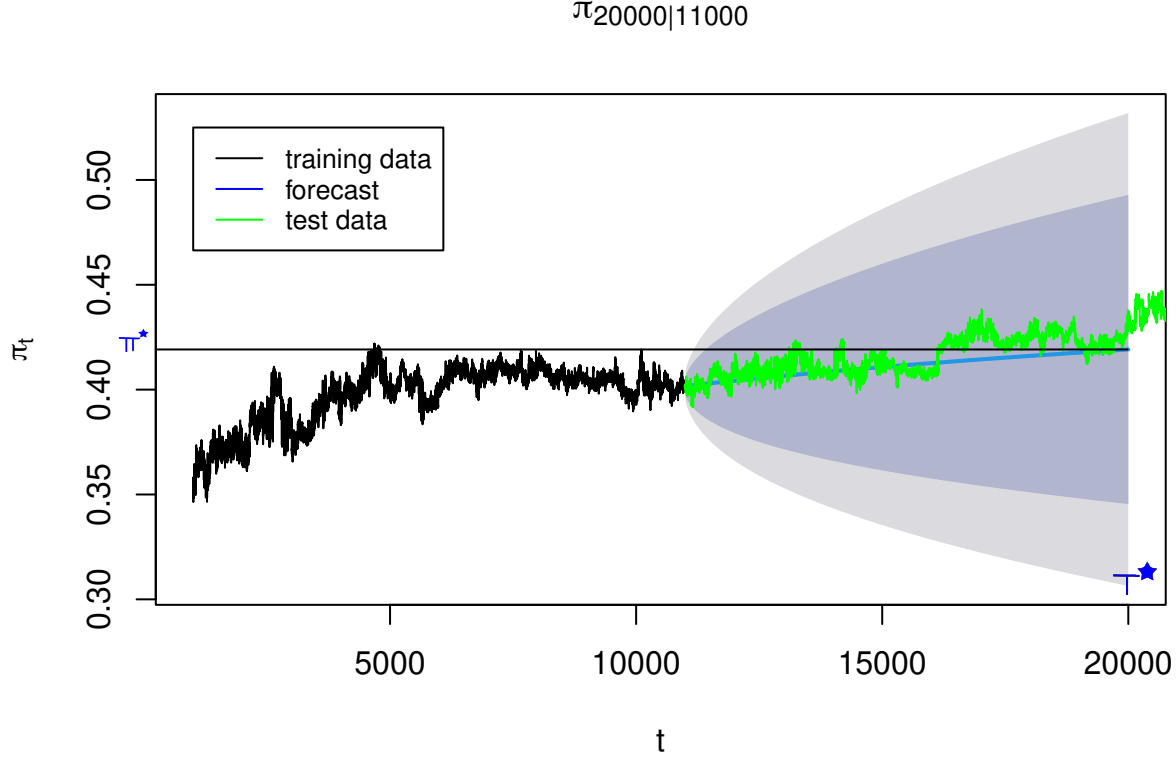


Figure 12: Estimated critical time \hat{T}^* versus security threshold π^* .

9.3 Limitations and future directions

The wide prediction intervals indicate substantial forecast uncertainty. This may stem from model misspecification or from genuine stochasticity in the adversary’s learning process. As discussed in Section 8, incorporating covariates (information content of queries, side-channel events) could reduce uncertainty and improve forecast accuracy.

Alternative model formulations (Gompertz curves, logistic growth models) that explicitly impose asymptotic bounds merit investigation. Such models would better respect the theoretical constraint $\pi_t \in [0, 1]$ and might yield more realistic long-term forecasts.

Despite these limitations, our analysis demonstrates that time series methods can provide quantitative guidance for encrypted search security policies, enabling data-driven decisions about system reinitialization timing.

References

- [1] Bias–variance tradeoff, April 2021. Page Version ID: 1018809373. URL: https://en.wikipedia.org/w/index.php?title=Bias_variance_tradeoff&oldid=1018809373.
- [2] Dickey–Fuller test, April 2021. Page Version ID: 1019701572. URL: https://en.wikipedia.org/w/index.php?title=Dickey_Fuller_test&oldid=1019701572.

- [3] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *Network and Distributed System Security Symposium (NDSS)*, 2012.
- [4] Hundman Rob. The ARIMAX model muddle, October 2010. URL: <https://robjhyndman.com/hyndsight/arimax/>.
- [5] Dawn Xiaoding Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *Proceedings 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pages 44–55. IEEE, 2000.