Thanks to Barnabé Monnot, Danny Ryan, and participants to the RIG Open Problem 3 (Rollup Economics) for helpful discussions and feedback.

EIP-4844 introduces two main components to the Ethereum protocol: (1) a new transaction format for “blob-carrying transactions” and (2) a new data gas

fee market used to price this type of transactions. This analysis focuses mainly on the data gas fee market and its key parameters

.

We first discuss the relation between the 4844 and 1559 fee mechanisms and their interaction. Then we present an analysis rooted in a simulation and backtest with historical data. Finally we discuss potential improvements on current setup.

1559 & 4844: a dual market for transactions

The data gas fee mechanism in 4844 is rooted in the 1559 mechanism. It introduces a similar adaptive rule, sustainable target, and block limit structure with minor differences that we summarize in the next section. The most important innovation is the fact that it is the first step towards multi-dimensional resource pricing in Ethereum (Endgame 1559?!). The blob data resource is unbundled from the standard gas

metering and gets its own dynamic price based on blob supply/demand

.

However, it is important to note that this is only a partial unbundling of the data resource. Standard transactions are still priced as before, with standard conversions for calldata of 16 gas units per byte and 4 units per empty byte. Only blob transactions use both markets with their EVM operations being priced in standard gas and their blob data being priced in data gas.

We essentially have a dual market for transactions with standard transactions using the one-dimensional (1559) mechanism and blob transactions using the two-dimensional (1559 x 4844) mechanism

. This distinction is important because users can decide to use any of the two transaction types and there will be relevant interactions between the two markets.

Data gas accounting & fee update in 4844

We give a summary that clarifies the key features of the 4844 data gas fee market. See relevant sections in the EIP-4844 Spec for details and Tim Roughgarden’s EIP-1559 analysis for an extended summary & analysis of the related 1559 mechansim.

Blob-carrying transactions have all fields required by the 1559 mechanism (max_fee

, max_priority_fee

, gas_used

) and also a new field (max_fee_per_data_gas

) to specify willingness to pay for data gas. This type of transactions can carry up to two blobs of 125kb each and these determine the amount of data_gas_used

which is measured in bytes. Such a transaction is valid if it passes the 1559 validity conditions & additionally the max_fee_per_data_gas

is bigger or equal to the prevailing data_gas_price

. Initially, the TARGET_DATA_GAS_PER_BLOCK

is set to 250kb (2 blobs) and the MAX_DATA_GAS_PER_BLOCK

to 500kb.

The data gas price for slot n

is computed with a beautiful formula

p^{\text{data}}n = m \cdot\exp\left(\frac{E[n-1}}{s}\right),

where m

is the MIN_DATA_GASPRICE

, s

is the DATA_GASPRICE_UPDATE_FRACTION

which corresponds to a maximum update of 12.5% up or down in consecutive blocks, and E_{n-1}

is the total excess data gas accumulated to date above the budgeted target data gas.

Backtesting with actual demand from Arbitrum & Optimism

Main 5 takeaways:

1. L2 demand structure is very different from user demand. L2s operate a resource-intensive business on Ethereum: they post transactions via bots at constant cadence, their demand is inelastic, and the L1 cost is their main operating cost.
2. Projected demand for blobs data from L2s is currently 10x lower than the sustainable target and will take 1-2 years to reach that level.
3. Until sustainable target demand is reached, the data gas price will stay close to the minimum (see chart and cold-start section).
4. When sustainable target demand is reached, the data gas price increases exponentially. If the minimu price is 1 this results in a 10 orders of magnitude cost increase for L2s in a matter of hours.
5. We discuss a few potential improvements that involve only minimal changes to the fee market parameters (see “Wat do?” section).

Let’s dive in…

We did a backtest using the historical batch-data load of Arbitrum and Optimism (this represents ~98% percent of the total calldata consumed by L2 batches). We chose the day with the highest batch-data load in the first two months of 2023, February 24th.

- Arbitrum posted 1055 consistent batches of ~99kb (600b stdev) every 6.78 blocks on avg, for a total of ~100mb/day
- Optimism posted 2981 variable batches of avg ~31kb (19kb stdev) every 2.37 blocks on avg, for a total of ~93mb/day

Analysis 1: fee update in practice

Arbitrum demand for data comes at a rate of 14.6kb/block and Optimism demand at 13kb/block (Ethereum blocks). With the current setup of the fee update mechanism, price discovery effectively does not start until the data demand load is above the block target of ~250kb

, about 10x the current load. To see this you can either assume that batch posters are smart in balancing and coordinating so that no block is above the target and the data price never moves from the initial minimum value of 1 wei, or simply notice that local increases in excess gas will quickly be absorbed so that the data price never goes much above 1.

[

1106×409 50.8 KB

](https://ethresear.ch/uploads/default/original/2X/7/7bf816afe78d4730a680859eb42486dddef45fd5.png)

EIP-4844 data gas price dynamics in two backtest scenarios: blob data demand rate is similar to historical (left) and blob data demand rate is 10x historical (right). Both assume same distribution for max_fee_per_data_gas

, uniform centered around 50 gwei (see link at the top for full simulation).

When potential demand is above the target, the price gets updated so that the blob transactions with lowest willingness to pay are dropped and balance with the sustainable target is maintained. But how long will it take? From January 2022 to December 2022 the combined data demand of Ethereum rollups increased 4.4x which means that continuing at this rate (or slightly higher considering innovation and 4844 cost reduction), it will take in the order of 1.5 years for the data price discovery mechanism kick-in and data price to start raising above 1

.

Having such a long time with data price at 1 creates unreasonable expectations on blob data costs

. Users and apps on L1 may make adjustments to start using blobs, only to be forced back to type 2 transactions once L2 demand (with higher willingness to pay) raises above target. In the worst case, this severe underpricing may lead to taproot-like usecases that will inject instability in the blob-data market that is undesirable from a rollup-economics perspective (as the next sections clarify).

Analysis 2: L2 costs

An Arbitrum transaction that posts one batch today consumes about 1.89M gas. Assuming an average fee of 29 gwei and an ETH price of $1500 the cost is about $85 per batch. The gas spent for calldata is about 1.59M and costs $70.

Switching to blobs will consume about 50K gas in precompiles at $2.2 cost, and 125K datagas which at a data price of 1 will cost $2e-10 (basically free

).

If the datagas price was, say 30 gwei, this would correspond to a data cost of $5.62 per blob, which is still 12x cheaper than the cost incurred for batch calldata today. This seems a price that rollups would be willing to pay and that is also fair, considering that the eventual sustained load on the system is 12x for calldata vs blobdata

:

“The sustained load of this EIP is much lower than alternatives that reduce calldata costs […]. This makes it easier to implement a policy that these blobs should be deleted after e.g. 30-60 days, a much shorter delay compared to proposed one-year rotation times for execution payload history.”

(EIP-4844)

Moreover, when starting at a price of 1, once demand hits the target data price will go up multiplicatively every 12s until demand starts dropping; considering that rollup demand is inelastic at such low prices, their costs will go up by 10 orders of magnitudes in a few hours. The 2022 FED rate hikes are a joke compared to this!

Cold-start problem

With the introduction of blobs we are bootstrapping a new market that will allow for price discovery and load balancing for blob data. As is always the case with new markets there is a cold-start problem, an initial phase in which we won’t have a strong market feedback and the fee mechanism will be in stand-by. This is even more relevant in our case because we are starting with a target much higher than current potential demand (even if all rollups were to switch overnight to blobs we will still be in oversupply).

Under the current setup, the analyses above highlighted a few problems:

(1) cold-start phase will be long (1-1.5 years)

;

(2) a price of 1 will incentivize spammy applications to use blobs

;

(3) a price of 1 for an extended period of time will set wrong expectations

: rollups and other apps may make assumptions based on the low prices only to be driven out of the market or seeing their costs raise by orders of magnitude in a matter of hours.

Wat do?

Idea 1: set a higher minimum price

The price of 30e9 considered in analysis 2 above is in the order of magnitude of the minimum price of 10e9 that was suggested in the EIP PR-5862 by Dankrad. Setting such a minimum price will set correct expectations, disincentivize spam, and is a fair price at current conditions

. It will make the cold-start phase not shorter but less problematic and it can be easily updated down/removed as the market warms up.

The PR was subsequently closed after pushback based on the argument that if ETH value goes up 1000x that minimum price will be too expensive & we should not make a design decision that will require an update in the future. I believe this is not enough of a good reason to rebut the proposed choice. We should weigh the benefits from having a higher MIN_DATA_GASPRICE

during the cold-start phase with the cost of having to update this param in the future

.

Ethereum is not-yet-ossified and there are many design choices that will be upgraded in the coming months/years: from small things like the conversion rates of different opcodes to gas to bigger things like ePBS. Considering the planned changes this one seems rather small.

Idea 2: set a lower block target for data gas

Another simple idea is to decrease the TARGET_DATA_GAS_PER_BLOCK

to one blob per block. This is still ~5x higher than current load, it will not solve spam and wrong-expectations in the cold start phase, but it will cut the cold start phase in half

.

It is a more cautious choice that can/will later be relaxed.

Idea 3: do nothing

Doing nothing will maintain all the problems with cold-start highlighted above. Playing devils advocate, it is possible that spam use of blobs at low prices will self-correct and make the cold-start phase shorter. But there is uncertainty on if/when this will happen and also higher induced volatility on data gas price for L2 businesses.

Open question

Should we consider changing the DATA_GASPRICE_UPDATE_FRACTION

currently corresponding to a maximum multiplier of 12.5% like EIP-1559? This could be a forward-compatible and less controversial change. It will not solve the problems above but, if decreased, could provide more stable prices over time.