Problem set 5: Atomicity and MEV

1 MEV in CFMMs

We will examine one of the common MEV techniques that takes advantage of atomicity: sandwiches. This type of MEV transaction is called 'sandwiching' since it proceeds as follows: a user attempts to trade δ of token B for some amount of token A, a 'sandwicher' then adds a trade before, which buys some amount of A (pushing the price of A up), and a trade after the user's, which sells however much of A was received from the first purchase.

We will analyze a sandwich attack against a user attempting to purchase λ units of token A with δ units of token B using a single 2-token CFMM with trading function φ. Recall that a trade of δ B for λ of A is valid if

$$\varphi(R_A - \lambda,\, R_B + \delta) \geq \varphi(R_A, R_B).$$

We will assume trades are reasonable, so this inequality is always saturated. For the rest of this problem, we will consider a Uniswap v2 swap pool, which has trading function $\varphi(R_A, R_B) = \sqrt{R_A R_B}$ (we assume no fees for this problem).

a) It is often more convenient to work with the *forward exchange function* $G(\delta)$ which specifies the amount of output token received λ for a fixed value of the input token δ for a swap $A \to B$. Derive an expression for $G(\delta)$ for Uniswap in terms of R_A, R_B, and δ.

b) Another useful function is the *price impact function* $g(\delta)$, which denotes the new price of A in terms of B, after a trade of δ. Derive the price impact function for this φ.

   Hint. Recall that the unscaled prices are given by $\nabla\varphi(R_A, R_B)$. Scaling everything in terms of B then means that the price of A for B is:

$$\frac{\frac{\partial}{\partial R_A}\varphi(R_A, R_B)}{\frac{\partial}{\partial R_B}\varphi(R_A, R_B)}$$

   at reserves R_A, R_B.

c) Show that $g(\delta) = 1/G'(\delta)$.

When users submit this trade, they do not know which other transactions will be placed in the next block. The reserves R_A and R_B may change from their observed values at the time of transaction submission, resulting in a different amount of output token received for a given input token amount. As a result, users specify a *slippage tolerance* η such that a trade is only executed if the user receives at least $1-\eta$ of the quoted amount quantity. For example, if a trade δ' is executed before the user's trade δ, the user will receive $G(\delta + \delta') - G(\delta')$ of token A instead of $G(\delta)$. Thus, the user's trade is only executed if

$$G(\delta + \delta') - G(\delta') \geq (1 - \eta)G(\delta). \tag{1}$$

A sandwich attack exploits this to extract as much value as possible by making the user receive the worst possible trade.

d) Calculate the δ' such that (1) is tight. We will denote this δ' as δ^{sand}.

e) After the user makes their trade of size δ, the sandwicher sells their δ^{sand} of token A back to the CFMM. Compute the profit from this trade, denominated in token B, in terms of δ^{sand}.