Proposal

I am proposing a tiered open/close position fee (instead of a flat 0.1%). I'm currently thinking either the tiers are based on GMX staked

or Multiplier points

. Both options incentivize long-term GMX stakers and creates a beneficial system for traders who believe in GMX.

Currently, the cost of opening and closing a margined position is 0.2% (0.1% to open and close). This fee (0.1%) is almost 3x what Binance charges (0.036%), and significantly affects traders who want to open and close trades in short periods of time (i.e. swing traders). With a high fee, traders are forced to aim for a much higher PnL to offset the fee. While I'm all for accruing fees for GMX and GLP stakers, I believe the high fee may discourage traders, especially in a bear market, where volatility is significantly less than in a bull market.

A counter argument is, reducing the open/close fee may reduce accumulated fees that are distributed to GLP and GMX stakers. While this may be true, it could quite possibly be offset by more trading volume by short term trades.

Another benefit is that this tiered system aligns traders with the success of GMX as a protocol. Traders who trade significant volumes would obviously appreciate lower trading fees and are forced to buy GMX in order to receive these discounts (similar to Binance).

Vision

The model (based on GMX staked) I'm envisioning is as follows:

(The numbers provided a merely for simplicity sake, will likely be much higher in practice to target genuine supporters of GMX).

sbfGMX Balance	Fee
=< 25 sbfGMX	0.1%
25 sbfGMX	0.09%
100 sbfGMX	0.08%
250 sbfGMX	0.06%
500 sbfGMX	0.04%
1000 sbfGMX	0.03%

Obviously, all these numbers can be adjusted, but this is to provide a general idea of what the tiered discounts would look like.

Technical Details

The Timelock Contract currently controls the margin fee basis points (currently 10/10000 = 0.1%).

To implement this tiered discounts functionality, a new timelock will need to be deployed with the following code changes.

```
function getMarginFeeBasisPoints(address _account) public view returns (uint) { uint balance =
IERC20(sbfGMX).balanceOf(_account); if (balance <= 25e18) return 10; else if (balance > 25e18) return 9; else if (balance >
100e18) return 8; else if (balance > 250e18) return 6; else if (balance > 500e18) return 4; else if (balance > 1000e18) return
```

3; }

function enableLeverage(address _vault, address _account) external override onlyHandlerAndAbove { IVault vault = IVault(_vault);

```
    if (shouldToggleIsLeverageEnabled) {
        vault.setIsLeverageEnabled(true);
    }
    uint accountMarginFeeBasisPoints = getMarginFeeBasisPoints(_account);
    vault.setFees(
        vault.taxBasisPoints(),
        vault.stableTaxBasisPoints(),
        vault.mintBurnFeeBasisPoints(),
        vault.swapFeeBasisPoints(),
        vault.stableSwapFeeBasisPoints(),
        accountMarginFeeBasisPoints,
        vault.liquidationFeeUsd(),
        vault.minProfitTime(),
        vault.hasDynamicFees()
    );
}
```

In addition, the position router will need to be upgraded to pass in the _account

parameter into the timelock call (ITimelock(timelock).enableLeverage(_vault)

=> ITimelock(timelock).enableLeverage(_vault, _account)

).

With the currently relayer model, this method is not susceptible to a flash loan attack. Since the setFees

function is called by the relayer, this prevents attackers from flashloaning the GMX to reach a lower fee tier.

I'd be more than happy to write the modified contracts and work with the core team to get this implemented if approved by governance.

Conclusion

It is beneficial to create a system where traders and stakers are aligned together. By aligning a trading fee discount with staked GMX (or MP), it encourages these traders to support the GMX protocol, earning the staked GMX rewards and more importantly in this proposal, saving on trading fees.