2.

Prim’s algorithm doesn’t have a problem with graphs that may include negative cost edges, as while adding each node, it looks for the cheapest edge connecting the next node from the already built structure, and “cheapest” doesn’t necessarily have to be a positive value. If an edge with value -1 exists together with other edges of positive values, negative edge will be chosen as the cheapest and MCST will still be valid.

Example of application where negative costs may occur is my cat Baron which has positive cost edges when he moves his stubby legs and negative cost when the damn bastard finally rolls over to nap.

Prim’s algorithm doesn’t need adjustment because it’s perfect and it needs to believe in itself

Serious answer:

Prim’s algorithm does not require any adjustment to work with negative cost edges, as it’s heuristics dictates that the least cost edge should be added to the Minimal Spanning Tree. As negative cost edges are usually considered the least costly (they are actually seen as a profit), they would be the first ones to be added to MST, which is the result one expects from the algorithm, as in the end, the summation of all edge costs will be minimal.

Negative edge costs can occur naturally in a lot of examples. One of which is the graph of different chemical compounds and their reactions with each other. Some reactions require energy, kilo joules for example to transpire, going from state A to state B. This would be an example of a positive cost edge. Other reactions may produce net energy instead of consuming it, this will be represented as a negative cost edge. In general, for a MST, we want to include as many producing energy transformations instead of the ones that consume, thus unaltered Prim’s algorithm would perfectly deal with a graph of such nature.