Artificial Hedging Intelligence: Part I

A Reinforcement Learning Approach to Hedge Fund Management

Written by: Velasquez, M. B.A., University of Houston.

Co-edited by: Willan, C. B.A., Oxford University, Ph.D., Stanford University.

July 2019.

PREFACE

Artificial Hedging Intelligence (AHI) is designed to manage a hedge fund. There are no disclosure documents or drafting attorneys necessary to foresee potential, contingent, or assumed risk factors that may result in unfavorable investment outcomes. There is no securities council, and any private placement memorandum is entirely in control of the users. There are no designated hedge fund attorneys or fund managers to navigate AHI investment strategies. All regulatory obligations are in control of the users. AHI funds and regulatory structures are in control of the users. Other needed service providers including administrators, prime brokers, and auditors, are in control of the users.

AHI provides accounting and back office operations. Auditing and due diligence items are public. Central (prime) broker operations are managed by reinforcement learning algorithms. Users are encouraged to act as *introducing brokers*. Regulatory filings such as Form D, investment advisor registration, and commodities registration, are in control of the users. AHI provides long and short equity strategies in equity and equity derivative securities. AHI uses fundamental and quantitative techniques to make investment decisions on behalf of the users. There is no pre-programmed bias for long funds or publicly traded equity and their derivatives. Lockups, gates, and other withdrawal functions are in control of the users.

AHI strategies use quantitative analysis for desired investment objectives. These strategies are constructed with reinforcement learning algorithms, econophysics, and combinations of mathematical and logical tools. AHI is not a *black box*. Everything is open source, and the technologies in AHI are available to users without proprietary protections. Global macro-strategies for AHI are guided by its reinforcement learning algorithms. AHI multi-strategy algorithms have high risk tolerance and prioritize capital preservation with several evolving core investment strategies. AHI users are in control of *accredited investor* standards and *qualified client* standards.

Broker-dealers and insurer operations are in control of the users. All AHI data is public. AHI offering documents protect the fund and its users. These documents and its terms are in control of the users. There are no management fees deducted from user accounts. Performance allocation is in control of the users. General overhead is in control of the users. High water marks under the *loss carry forward* terms are controlled by users. There are no hard or soft hurdle rates. AHI does not require minimum investments, thresholds, or conditions to invest.

AHI does not require initial lock-up periods. Lock-ups are shortened or lengthened at the user's discretion. AHI's strategy for illiquid investments does not require sponsors or closed-end private equity. There are no gate provisions in response to large redemption requests. Expenses of the fund, like legal formation costs, regulatory filings, brokerage

costs, or clearing costs, bear on the amortization periods decided and agreed upon by the users. AHI can operate equally well domestically and offshore. There is no master-feeder or side-by-side structure for separate investment portfolios in AHI. Blue Sky Laws and other exemptions for securities registration are controlled by the users. Registering or exempting from the Commodities Exchange Act is controlled by the user. AHI reinforcement learning algorithms function as commodity pool operators and commodity trading advisors. Registering exemptions for AHI with the Commodity Futures Trading Commission and registering AHI as a member of the National Futures Association is in control of the users.

INTRODUCTION

A computer program that creates hedging strategies has three levels of ignorance: either it knows the rules and has information; it knows the rules and lacks information, or it doesn't know the rules.

The semantics and pragmatics of programs are indispensable for constructing sensible norms within a hedging strategy. The successful assumptions made in one hedging strategy help determine the normative models throughout the program. Consistent hedging cannot be reduced solely to the mechanical insertions of numbers into formulae without contextually sensitive normative models. In fact, teaching a program to hedge with axioms does not help it discover normative strategies or theorems.

Fundamentally, hedging strategies are systems that attempt to calculate order in chaos. The *representation* of numerical information in AHI is designed for hedging strategies. *Credence* functions make the assumptions for strategies to determine *why* information enters its algorithmic calculations.

However, there's a problem. Probability theory is *mute* on information representation. Probabilities and percentages were invented as a means of representing information but is improbable that computer algorithms will continue to use probabilities and percentages rather then modeling them through complex geometry.

For example, as programs become incrementally *self-aware* through sensors, IoT, and new architectures, hedging strategies will inevitably acquire data through frequencies *as experienced*. AHI uses geometry to define the discrete mathematical relationships between logic and probability, *as experienced*.

Currently, only genetic algorithms are capable of learning the preparatory steps for qualitative approaches to hedging. AHI understands why markets crash, when crashes begin and end, and how to spot bubbles to later predict them. It uses physics as a *bridge* between geometry and algebra. This is the combination of disciplines that effectively code the *emergent* and *reductive* properties of both probability and possibility theory.

The goal of using reinforcement learning to manage a hedge fund is to define the geometry of markets. A reinforcement learning algorithm interacts with the market environment and, upon observing the consequences of its trades, alters its algorithmic behavior in response to profit and loss. Through trial and error, reinforcement learning algorithms eventually gain optimal control of the mathematical formalisms underpinning successful strategies. Optimal policies are defined by the program through optimal policy aggregations over time with strong temporal correlations between trades. AHI is designed to explain why new market discoveries

come from different formulations of the same physical laws, even if they are mathematically equivalent.

Anyone who has studied the deterioration of precision over the course of a long calculation knows that deterioration is due to two things: the accumulation of errors by superposition and the amplification of errors committed early in a calculation. Most operations performed in algorithmic schemes attempt to hedge against the deterioration of precision without a durable foundation in logical, physical, or mathematical principles. Econophysics provides AHI with the best analytical framework with which to map the predictability, possibility, and probability of market changes over time without resulting in chaos or noise. Increasing the complexity and speed of algorithms does not decrease the enormous magnitude of the assumptions the algorithms make. There is no evidence to suggest that increasing mathematical variabilities produces efficiency. In fact, the *race to zero* in high frequency trading only adds more noise to the market.

AHI provides an understanding of outcomes; not a collection of answers. With reinforcement learning algorithms, AHI learns to free itself from users' persistent belief that coordinates must have a metrical meaning. This results in geometry that encourages spatial temporal difference learning within the program and cogently maps the relationships between geometry and algebra.

The following desiderata are a guide to an interdisciplinary approach for artificial intelligence in hedge fund management.

DESIDERATA FOR PROGRAMMING AHI

Desideratum I.

A reinforcement learning (RL) hedging algorithm is constructed for the control and execution of a complicated trade, and possesses a purely logical part and a purely arithmetical part. This is because defining information without immediate recourse to formulae and numbers results in a qualitative combination of circumstances and events. These results are rendered by periodic frequency functions. Thus, the scaling effects of precision are geometrically defined as the program learns to trade.

Desideratum II.

Data is described in a RL hedging algorithm as a complex system of properties that emerge from the interactions of its individual components. These components rarely have complete information and cannot be ubiquitously *rational*.

Desideratum III.

RL hedging algorithms are analogous to properties of the physical world, such as thermodynamics, and to statistical mechanics. They are applied to assemblies of large trades.

Desideratum IV.

RL hedging algorithms find correlations that allow the market behavior of collectors to evolve over time. These correlations do not have initial conditions.

Desideratum V.

RL hedging algorithms analyze the random variability in markets to distinguish between the random variables that appear in the distribution functions. The values of random variables represent the geometric outcome of events.

Desideratum VI.

RL hedging algorithms are flexible enough so that increments of time need not refer to fixed intervals, but to arbitrary successive stages of trade decisions; both creating and optimizing algorithmic steps and higher-level rules. The program assumes that all moments in time are finite, and moreover that neglecting higher order terms in a series expansion results in central limits. The one-step dynamics of trades lead to probabilities that reduce markets to a closed-form Chapman-Kolmogorov integral equation.

Desideratum VII.

Hedging strategies that use *greedy* algorithms to search for optimal trades, based on *immediate* market data, are prevented from future access to alternative trades. A Langevin

approach to modeling market data with Brownian motion counterbalances and enhances future access to alternative trading strategies.

Desideratum VIII.

There are market processes where the volatility, $\sigma^2(t)$, varies as a fractional power of time. Optimal RL hedging strategies are quasi-stochastic; meaning, the algorithms trade as a function of market *phases*. Computing optimal learning policies, given environmental models of Brownian motion, uses continuous trades. Asynchronous hedging strategies are easier to intermix with real-time interactions. RL hedging algorithms experience Brownian motion to determine what dynamic programming applies to its backups. The *latest* value for trading policies guides the overall dynamic programming; making *relevance* possible in algorithmic decision making. The value functions of a trade stabilize when they are consistent with current policies. Policies stabilize when they are *greedy* to the current value function.

Desideratum IX.

Measurements and assumptions of log prices and random variables are mapped to upper and lower bounds of magnitude errors in hedging strategies. Algorithms use each policy evaluation to assure that strategic bounds are sufficiently small. This fosters a convergence in the evolution of probability distributions for trades in Brownian motion.

Desideratum X.

RL hedging strategies *normalize* the variation of autocorrelation functions that decay over time spans of thirty minutes or less, with temporal difference learning that trade without models of market dynamics. This strategy updates its estimates based on other learned estimates and bootstraps Fokker-Planck partial differential equations that use n-th order diffusion constants to compute changes in the average of random variables.

Desideratum XI.

RL hedging algorithms assume that market fluctuations are fast and random over the small time interval Δt; while the average return increases linearly or periodically with time. The existence of non-zero diffusion coefficients computes numerical values using empirical data for log prices and return values. A fixed policy condition that uses temporal difference learning, defines convergence in the mean of constant small step size parameters that decrease in accordance with stochastic approximation conditions. The differences in optimal rates are measured and used to predict non-zero coefficients with algorithms that *experience* trades until an optimal strategy is converged upon. For every time step t, nonterminal states visited by the algorithm change the value function once. *Experiences* are then processed again with a new value function to produce newer *experiences* until the value function converges in a batch updating manner.

Desideratum XII.

RL hedging strategies that perform batch updates with a temporal difference of zero, converge deterministically to the step size parameter α; if it is small enough. The program differentiates between the average return and the average price as two growth rates that yield a minimized mean squared error. The algorithms define estimates that would be correct for the maximum-likelihood model of a Markov process.

Desideratum XIII.

RL hedging algorithms use *average* one step and *infinite* step back functions for hedging strategies. These strategies interrelate temporal difference learning and Monte Carlo algorithms to asset price fluctuations by the error reduction properties of n-step returns. In the program, this yields a combination of experience-based and model-based trading. A mechanistic approach to temporal difference learning provides incremental approximations for a successful eligibility traces; offline or backwards in geometric time allowing errors to be assigned a priori.

Desideratum XIV.

RL hedging algorithms use time dependent functions to define density distributions that provide the *characterization* for both the long and short times of autocovariance functions. The program uses Fokker-Planck equations to account for the fat tails in its distribution functions of option prices.

Desideratum XV.

RL hedging algorithms use conditional ensemble averages to define time averages. For example, *small-time* decays square correlations for financial data more slowly than do linear correlations. RL algorithms take the *long-time* square correlations that exhibit slow power law decay. They also use the maximum-likelihood estimation of decay to parameterize trade values. RL hedging strategies provide models for the algorithms to compute estimations of decay rate value functions. Certainty-equivalence estimates assume the underlying power law has high levels of certainty.

Desideratum XVI.

RL hedging algorithms define the skewness of data as a *characteristic* of financial data. The algorithms build temporary records of trades and use traces to mark the memory parameters associated with learning changes and temporal difference errors.

Desideratum XVII.

RL hedging algorithms assume that volatility scales with the square root of time. Those trading strategies which are executed and/or revisited before the first trade decay to zero.

Desideratum XVIII.

RL hedging algorithms have one estimate for a fair value trade and one estimate for the future. Rising trends add to the demand of an asset and falling trends reduce demand in future estimates.

Desideratum XIX.

Market trends that lead to a biased random walk, are larger than the negative threshold in qualitative trading. The program assumes that asset prices fluctuate around an initial value. After a series of trade jumps, returns reach the local maximum and cross into the larger negative values of finite time.

Desideratum XX.

RL hedging algorithms assume that as prices go up, the percentage of purchases made by investors increases. When profit incentives disappear, the program analyzes past patterns.

Desideratum XXI.

Given the state of a market and a successful trade, forecasts in the program predict the results of future trades. The algorithms approach market analysis stochastically and distribute trade possibilities and their probabilities stochastically.

Desideratum XXII.

RL hedging algorithms perform state space planning to find the optimal trading policy. Trading causes transitions from state to state while value functions use evolutionary algorithms and partial order planning to plan space methods. This applies to the stochastic optimal control problem in every strategy.

Desideratum XXIII.

RL hedging algorithms plan online. New data changes the strategy and customizes the online planning process.

Desideratum XXIV.

Within a strategy, there are two learning *experiences*: model learning and direct reinforcement learning. Both *experiences* propagate backward, grow fast, and produce trading pairs. The program defines market environments as probabilities. The weight of a probability contributes to the urgency of a trade. RL hedging strategies prioritize trades according to the measure of their urgency and execute trades with prioritized sweeping. A queue maintains every trade with an estimated value that will change. When the top trade is executed, the probability

threshold is surpassed and the trade is placed in the queue with new priorities. The effects are propagated backward until *quiescence* properly defines a sample plan.

Desideratum XXV.

The difference between a hedging strategy and the market environment is that any given strategy has consequential trades with limited probabilities. Sample trades and their returns are modeled by current policies in the algorithms.

Desideratum XXVI.

In a uniform hedging strategy, algorithms cycle through successful trades while backing up on-policy simulations that occurred under *greedy* policies.

Desideratum XXVII.

RL hedging algorithms use heuristics to improve trading. The current value function of each trade is represented by a tree that approximates leaf nodes and roots. When the value of a node is computed, the best trade is chosen.

Desideratum XXVIII.

If there is a *perfect* strategy for an imperfect value, deeper searches at the end of an algorithm are eliminated and search trees focus on the trades that follow efficiency.

Desideratum XXIX.

RL hedging algorithms that hedge against changes in the market will encounter trades never experienced before. Trade spaces in the algorithms use continuous variables to learn from previous trades. This prediction method describes the estimated value of a trade.

Desideratum XXX.

RL hedging algorithms with on-policy distributions use trading frequency to select trades according to values approximated over the minimization of errors.

Desideratum XXXI.

Trades are continuous and two dimensional with a vector of two real components. RL hedging algorithms initialize from one point to another by size. Trades are represented with width to influence the density of trades for online tile coding.

Desideratum XXXII.

In RL hedging algorithms, radial basis functions over binary features produce approximations that smooth and differentiate non-linear changes at the center of widths. The algorithms learn scalar signals to define trade *critiques*.

Desideratum XXXIII.

RL hedging algorithms assume that fluctuations in markets arise from the *motion* of prices. The algorithms also assume that exchanges between trades are represented as ensemble averages.The algorithms define market phases as the interatomic interactions of proper statistical averages at specific times.

Desideratum XXXIV.

Intelligent mapping is distinguished by points of *metastability* and the ability to develop a forecast for monopolies. Intelligent network *motifs* uncover the geometric structure of complex trade networks with positive feed forward loops. The algorithms within each strategy discretely model market regulations with *logical* functions. These models list the value of the variables in the starting vectors by calculating rate constants, partial differential equations, differential equations, and parameter problems for each trade. A system progression is defined during the analysis phase of solved equations. Each strategy investigates the behavior of the market and the variables for the parameters of its analysis; resulting in vector fields that determine changes *where* the rate of trajectory is successful.

Desideratum XXXV.

Intelligent bifurcation maps handle large variables and parameters for trades. Trades are represented by points on a map and once the parameters pass the threshold value, qualitative changes occur in the space of irreversible checkpoints. The algorithms use switch controls to produce different output values for the same input values; depending on the behavioral history of inputs.

Desideratum XXXVI.

In RL hedging algorithms, the object rules for applications produce new sets of outputs that prevent rules from prioritizing; allowing competition, fitness, and dominance.

Desideratum XXXVII.

Maps are designed through training where trade predictions are corrected when necessary. Maps are designed with the desired level of dimensionality and accuracy. These maps contain deducible geometry derived from the strategies of optimal trades.

Desideratum XXXVIII.

The independent variable of one algorithm is the dependent variable of another; thus, there are no *genuinely* independent variables in circular or recursive strategies.

Desideratum XXXIX.

RL hedging algorithms assumes risk when multiple trades and their probabilities are *known*.

Desideratum XL.

RL hedging strategies assume Knightian uncertainty when the probabilities for future trades are *known*.

Desideratum XLI.

Market prices of financial assets do not reflect their fundamental value; but rather, expectation. RL hedging algorithms assume the discounted present value of future earnings, diverge.

Desideratum XLII.

RL hedging strategies change to match the reality of the market. The algorithms make trades to match self-correcting negative feedback loops and self-reinforcing positive feedback loops.

Desideratum XLIII.

RL hedging algorithms assume that currency markets have *symmetrical* upsides and downsides without equilibrium.

Desideratum XLIV.

RL hedging strategies do not assume that price deviations from a putative and random equilibrium, break down.

Desideratum XLV.

RL hedging algorithms never analyze data with dichotomies owing to the violent transitions they exhibit.

Desideratum XLVI.

RL hedging algorithms produce *reflexive* feedback loops for non-recurring changes and other statistical regularities.

Desideratum XLVII.

RL hedging strategies assume that probability stems from a geometric spectrum of artificial and natural domains.

Desideratum XLVIII.

RL hedging strategies have internal models that enable algorithms to *move* from an analysis function to a manipulative function. The internal model updates and changes in response to trades. In this way, feedback connects market analysis to future trading.

Desideratum XLIX.

RL hedging algorithms assume strategies are complex due to interactive trade optimization functions and feedback non-linearity.

Desideratum L.

RL hedging strategies learn from internal models to improve algorithmic performance.

Desideratum LI.

RL hedging algorithms assume trades are sensitive to both conditions of the market and path dependence; thus, predicting the future path of a trade with program rules, parameters, and initial trade conditions, only functions in strategies with finite limits of market data.

Desideratum LII.

RL hedging strategies define market uncertainties and indeterminacies as inherently contingent and time bound to *new* regularities and properties, and therefore as being plausibly emergent at higher levels of aggregation.

Desideratum LIII.

RL hedging algorithms explain time bound and contingent market regularities through analyses of verifiable patterns.

Desideratum LIV.

RL hedging strategies define evolutionary processes with *credence* functions that analyze market *limits*.

Desideratum LV.

RL hedging strategies define markets as both a deterministic and random. This permits strategic experimentation.

Desideratum LVI.

RL hedging strategies never separate raw data from algorithmic models. Rather, the algorithms fit specific markets with goals of falsifying strategies.

Desideratum LVII.

RL hedging algorithms attempt to *define* inherently imperfect models to improve analytical, statistical, computational, qualitative, and historical functions; allowing a *strategic belief function* to be self-fulfilling and self-reinforcing.

Desideratum LVIII.

RL hedging strategies define trade outcomes to increase *confidence* in algorithmic models; leading through *widespread* credence functions.

Desideratum LIX.

A combination of hedging strategies designed for uncertainty, assume that heuristics reduce the cost of computation; which includes lump adjustments and trade reevaluations associated with *punctuated* market equilibria.

Desideratum LX.

RL hedging algorithms define *foresight* for trades without preconditions for market characteristics. *Correct* foresight in a market is impossible because there are no trade flows or direction.

Desideratum LXI.

RL hedging strategies support multiple internal configurations corresponding to different algorithms.

Desideratum LXII.

In RL hedging strategies, the credence function analyzes market data to determine the parameters for the algorithms.

Desideratum LXIII.

RL hedging strategies assume markets have the *appearance* of *normality* because feedback loops only modify trades that are not distributed.

Desideratum LXIV.

RL hedging strategies define market impacts as the square root of order size and change in price.

Desideratum LXV.

RL hedging strategies define *continuous double auction* as the mechanical response to strong constraints.

Desideratum LXVI.

RL hedging algorithms use *fallibility functions* that include both probabilistic risk and Knightian uncertainty.

Desideratum LXVII.

RL hedging algorithms combine probabilistic risk and Knightian uncertainty to define credence functions. Potential values for the discrepancies between algorithmic outcomes and overarching probabilistic rules, allow *extant* credence functions for rational expectations, behavioral finance, and imperfect knowledge economics.

Desideratum LXVIII.

RL hedging algorithms assume that the mathematical modeling of social processes changes trade outcomes in ways that do not conform to an overarching probabilistic rule.

Desideratum LXIX.

RL hedging strategies mathematically formalize imperfect understanding; exposing models to the growth of data driving optimal trades. The algorithms explore the possibilities of trade behavior and define *dependency* therein.

Desideratum LXX.

RL hedging strategies jettison the *determinate* restrictions of macroeconomic forecasts that change over time; recognizing that dependence on data necessarily revises strategies. When restrictions on revisions are imposed, it produces data that is compatible with *causality*.

Desideratum LXXI.

RL hedging strategies define trade positions between unrestricted models and determinate models with an *intermediary* credence function.

Desideratum LXXII.

RL hedging algorithms generate empirical data for trades that protract estimations over time while *revising* the movements and causal factors of market regularities; formalizing credence functions with Knightian uncertainty and time series evidence.

Desideratum LXXIII.

RL hedging strategies assume that *imperfect* economic models are contingent on the swings of asset price and risk. These conditions in macroeconomic and financial models define irregular asset prices.

Desideratum LXXIV.

RL hedging strategies assume expectation models represent conditional rationality in macroeconomics and finance. The qualitative terms that *synthesize* the underpinnings of a forecast, are probabilistic predictions.

Desideratum LXXV.

RL hedging strategies assume that time- and context- bound trades yield specific explanations for successful trades; rather than timeless and universalizable generalizations.

Desideratum LXXVI.

RL hedging algorithms assume that when analytical functions operate at the same time they deprive themselves of independent variables. Thus, neither function has a *genuinely* independent variable.

Desideratum LXXVII.

RL hedging algorithms use propagation functions to learn the existence and location of *tipping points* for extreme market conditions. Strategic credence functions define market equilibrium or *near* market equilibrium, as normative and persistent behavior.

Desideratum LXXVIII.

RL hedging strategies assume rational expectation models depend aggregations of individual trades, correlated and modified adjust for the errors of large dependent variables.

Desideratum LXXIX.

RL hedging strategies assume rational expectation models produce a constant price to dividend ratio.

Desideratum LXXX.

RL hedging strategies assume asset price models have positive feedback when *actual* price deviation increases or decreases. The *average* deviation also increases or decreases.

Desideratum LXXXI.

RL hedging strategies update both *fundamentalist* credence functions and *chartist* credence functions with multinomial logit models. The relative performance of trades measure how quickly the hedging algorithms alternate between strategies.

Desideratum LXXXII.

RL hedging strategies assume strategy switching drives short run profitability. Self-fulfilling equilibria with endogenously generated bubbles, trigger shocks and fuel the

positive feedback from trend following; thus, allowing market crashes to reinforce negative feedback.

Desideratum LXXXIII.

RL hedging strategies predict prices with the weighted averages produced by its algorithms. Trend following rules extrapolate price changes with weak parameters or strong parameters. Anchor and adjustment rules extrapolate price changes from flexible anchors while forecasting heuristics evolve according to the discrete models using asynchronous updating.

Desideratum LXXXIV.

RL hedging strategies assume bounded rationality models of first order approximation use the Lucas critique for trade expectations and policy changes.

Desideratum LXXXV.

RL hedging strategies assume probability is limited to probabilistic errors that do not fall on bell-shaped curves.

Desideratum LXXXVI.

RL hedging strategies assume *reflexive* functions linked to frequency dependency compete and cooperate with, and are parasitic or symbiotic of, one another; allowing uncertainty to operate with both the *tempo* and *mode* of evolution.

Desideratum LXXXVII.

RL hedging algorithms assume reflexive functions that cross a threshold, do not remain stable enough to repeat. The repetition of the *same* boom and bust cycles never repeat.

Desideratum LXXXIII.

RL hedging strategies assume markets are composed of nested strategies that result in extremely brief local equilibria. Market changes are defined with iterations of unsynchronized combinations of trading strategies. When the rate of price change in markets increase, the *lifetimes* of local equilibria disappear.

Desideratum LXXXIX.

RL hedging strategies assume that reflexive functions are short-lived and cannot be exploited; thus, effective exploitation management *includes* long-lived and widespread equilibria.

Desideratum XC.

RL hedging algorithms assign probabilities to logical statements and refine their probabilities over time.

Desideratum XCI.

RL hedging strategies have logical inductors that predict patterns of falsehood in logical statements. Trading strategies in polynomial time use statistical summaries to predict trade sequences with *pseudo-random* truth values. The algorithms learn *new* credence functions from *current* credence functions to avoid self-reference.

Desideratum XCII.

RL hedging strategies assume logical inductors for future beliefs are strictly dominated by universal semi-measures.

Desideratum XCIII.

Logical induction criteria assume that there is no algorithm with finite risk tolerance.

Desideratum XCIV.

RL hedging algorithms assume probabilities do not define the uncertainty of *logical facts* in a strategy.

Desideratum XCV.

RL hedging algorithms define analysis functions in isolation as changes in credence; reducing uncertainty by analyzing trade π. Strategies experience both empirical uncertainty and logical uncertainty.

Desideratum XCVI.

Logical induction functions assign probabilities to trades that evolve over time through a *deductive* process.

Desideratum XCVII.

Logical inductors define credence as *consistent* as market time approaches infinity. Probability values for different patterns, outpace deductive functions.

Desideratum XCVIII.

Logical induction functions execute market analysis with empirical data and deductive parameters.

Desideratum XCIX.

RL hedging algorithms assume credence functions always aggregate the logics of longer algorithms.

Desideratum C.

RL hedging strategies are calibrated for *geometric* time. Analytic functions assign nonzero probabilities to Peano arithmetic algorithms and to conjunctive rules that converge to values above 0, despite infinite conjunctions.

Desideratum CI.

RL hedging algorithms do not define logical contradictions.

Desideratum CII.

RL hedging strategies assume that *market prices* are inconsistent valuations. The algorithms define *prices* as consistent with the logical uncertainty that develops credence.

Desideratum CIII.

RL hedging algorithms assume when logical inductors learn patterns, the pattern is the source of probabilistic estimates calibrated for trades. Thus, credence functions oscillate.

Desideratum CIV.

RL hedging algorithms assume there is no *holistic* method for detecting predictable bias with logical induction or credence functions.

Desideratum CV.

RL hedging algorithms mathematically formalize random analysis to define the deterministic digits of π.

Desideratum CVI.

RL hedging strategies assume Cromwell's rule for analytic functions does not define extreme probabilities unless data is logically true or false. The algorithms generalize Cromwell's rule when there are logical uncertainties that have not been proven or disproven. Thus, logical inductors use an infinite sequence predictor to analyze empirical uncertainty and logical uncertainty.

Desideratum CVII.

RL hedging algorithms assume credences about market logic are fixed enumerations of analysis and conditioning from Peano arithmetic rules. The probabilities that semi-measures assign to conjunctions, go to zero.

Desideratum CVIII.

RL hedging strategies assume high frequency trading never stops, making forecasts a diagonalization of self-reference functions.

Desideratum CIX.

RL hedging algorithms assume probability functions define future credence whenever there are changes in response to new market data. *Current* credence is defined by expectation functions and their weighted probabilities.

Desideratum CX.

Logical inductors update credence functions by discovering patterns in markets; with ongoing processes and deduction. Thus, the *self-trust* properties of logical inductors correlate *expectations* and different market patterns.

Desideratum CXI.

RL hedging algorithms assume logical inductors assign probabilities to *geometric* market patterns.

Desideratum CXII.

Logical inductors learn to write code for market scenarios where analytic functions are uncertain about the mathematical framework of statistical ensembles. This allows logical inductors to predict the aggregate of market *ineffectiveness*. The algorithms write code for trade models that integrate logical patterns and obey logical constraints.

Desideratum CXIII.

RL hedging strategies use *statistical* guarantees in settings where trade robustness and reliability are not defined.

Desideratum CXIV.

RL hedging algorithms analyze the behavior of the overall program with credence functions, heuristics, and approximations. This results in designs that model self-reference; allowing algorithms to assign probabilities for: logical uncertainties in markets, model trading behavior, and deductive limitations.

Desideratum CXV.

RL hedging algorithms are in *positions* of self-locating uncertainty during periods of market splits; via *market decoherence*. The outcomes of markets registered by analysis functions do not affect the probabilities assigned to local trades.

Desideratum CXVI.

RL hedging strategies assume markets emerge out of Everettian quantum mechanics. The algorithms learn their trading circumstances and the state of markets through the evolution of conditions of self-locating uncertainty. in which the algorithms cannot *fully* approximate which isolated market phase they inhabit.

Desideratum CXVII.

RL hedging algorithms define probability as fundamentally subjective. Probabilities are not written into code. Only a spontaneous collapse function that captures the degrees of *belief* from a credence function is assigned by the Born rule.

Desideratum CXVIII.

RL hedging algorithms give equal credence to discrete sets of trades that are consistent with market data.

Desideratum CXIX.

RL hedging algorithms determine the probabilities assigned to trades without changing the *rationality* that formalizes constraints on credence.

Desideratum CXX.

RL hedging strategies assume the phases of markets are defined by a universal wave function.

Desideratum CXXI.

RL hedging strategies assume that markets are given by a wave function. The evolution of time is always in accordance with Schrodinger's equation to amplify *microcosmic* superpositions; leading to trades where *macrocosmic* algorithms are also in superposition.

Desideratum CXXII.

RL hedging algorithms assume trades are defined by a reduced density matrix; generated with partial traces over Hilbert space.

Desideratum CXXIII.

RL hedging algorithms assume *market decoherence* diagonalizes the reduced density matrix for *macrocosmic* degrees of freedom. The algorithms measure with a pointer basis that yields one result.

Desideratum CXXIV.

RL hedging algorithms assign probabilities to the analytic functions that execute trades. The probabilities in self-locating uncertainty produce several copies of themselves; each in different *phases* of a wave function.

Desideratum CXXV.

RL hedging strategies define market *phases* as the *branch structures* of a wave function. Strategies are not only trading algorithms with dynamic rules, but also subjective coarse-graining statistical mechanisms.

Desideratum CXXVI.

RL hedging strategies assume self-locating uncertainty after market decoherence, register trades in a post-measurement pre-observation period; allowing the algorithms to *learn* what market phases their trades inhabit.

Desideratum CXXVII.

RL hedging algorithms assume when wave functions become sufficiently entangled and produce *market decoherence*, trading is experimental.

Desideratum CXXVIII.

RL hedging strategies assume algorithms in situations of self-locating uncertainty, do not learn and reorganize credence functions before learning trade outcomes. *Approximating* credence reconstructs the probabilities assigned during the post-measurement pre-observation period; making them relevant for deciding which credence is successful.

Desideratum CXXIX.

RL hedging strategies assume *indifference* functions for qualitative analysis have identical market data. This is not to be confused with the Laplacean principle of indifference.

Desideratum CXXX.

RL hedging algorithms define indifference functions as a constraint on *rational* probability. For example, the probability that trade H is successful, is given by the fraction of market phases in which trade H holds. The fraction of market phases is the same as the fraction of *success* in the indifference function.

Desideratum CXXXI.

RL hedging algorithms contain *epistemic separability* functions to separate the market from trades; thus, separating trades from credence functions. The epistemic separability function assumes market phase U contains a set of trading patterns S, such that every analytic function

with qualitatively identical data for trade A, is a pattern of element S. The probability of trade A in the trading pattern $X \in S$, given that both are in market phase U, contains trading pattern S.

Desideratum CXXXII.

RL hedging strategies assume markets are *decomposed* into phases. Each market phase is connected and used to reconstruct the *original* market. This allows the data of each phase to be categorized with or without the influence of evolution.

Desideratum CXXXIII.

RL hedging algorithms assign higher probabilities to trades that already occurred and lower probabilities to wave functions.

Desideratum CXXXIV.

RL hedging algorithms define vectors as the sum of orthogonal vectors with equal amplitudes. The algorithms unitarily transform market data to justify epistemic separability functions with equal credence in each vector, implying a *Born rule* way of counting.

Desideratum CXXXV.

RL hedging algorithms rotate, spatially translate, temporally shift, and conformally map the market to the probability of trade A in market phase $X \in S$. Both are in market U and market U is identical to $Mx(X) \in S'$. Given that in market U' a set of market phases S' exist, the possibility of trade A is an element of S'.

Desideratum CXXXVI.

RL hedging algorithms with epistemic separability functions produce indifference functions in cases of trade duplication.

Desideratum CXXXVII.

RL hedging algorithms assume in *microcosmic* trades, the Born rule is expressed as the probability of an outcome given by the *expectation* value of a trade projection operator.

Desideratum CXXXVIII.

RL hedging strategies assume the probability of a successful trade depends on probability amplitudes. Thus, the probability assigned to market phases is not proportional to the finite integral of $\eta(t)$ over time. Rather, the assignment is weighted by the amplitude $\alpha(t)$ to a corresponding phase.

Desideratum CXXXIX.

RL hedging strategies assumes that to prove a period of uncertainty of any length, treat ψ^2 as the weighted probability proportional to the limit of geometric time before zero.

Desideratum CXL.

RL hedging algorithms test wave functions by analyzing eigenvalues.

Desideratum CXLI.

RL hedging algorithms assume that the reduced density operator for trade A, does not define the success of trade A; thus, justifying the reduced density operator as a *statistical* measurement for trade A.

Desideratum CXLII.

RL hedging strategies do not assume that *reduced density matrices* are the only *viable* way to mathematically define markets.

Desideratum CXLIII.

RL hedging algorithms mathematically define markets as the connection between yield and evolution in isolation.

Desideratum CXLIV.

RL hedging strategies assume it is *irrational* to give stronger credence to conjunctions of trades, than to one of its conjuncts.

Desideratum CXLV.

RL hedging algorithms assume it is irrational that, ceteris paribus, one trade is more likely than another; when assigning stronger credence to the second trade rather than the first.

Desideratum CXLVI.

RL hedging algorithms assume it is irrational to plan trades *solely* based on learning the *propositions* of other trades.

Desideratum CXLVII.

RL hedging algorithms assume it is irrational that one trade is more likely than the other if there is no data that differentiates them.

Desideratum CXLVIII.

RL hedging algorithms define *doxastic* trades as a function that takes new trade propositions and returns real numbers that measure credence.

Desideratum CXLIX.

RL hedging algorithms assume 0 measures minimal credence and 1 measures maximal credence. Thus, a doxastic trade is defined by the function $c : \{A, B\} \rightarrow [0, 1]$. [0,1] is the set of real numbers of *at least* 0 and *at most* 1; {A,B} is the opinion set and c is the credence function.

Desideratum CL.

RL hedging algorithms define a *no drop rule* as the trade with opinion set {A,B}. A entails B and $c(A) \leq c(B)$. Thus, credence requires a *no drop* over *logical entailment functions*.

Desideratum CLI.

RL hedging algorithms use *rational* trade options in strategies that follow *unique* credence functions to satisfy market conditions.

Desideratum CLII.

RL hedging strategies assume the credence for one trade is more accurate than another if there are greater numbers of *true belief* functions than *false disbelief* functions.

Desideratum CLIII.

RL hedging algorithms define the accuracy of credence with proximities to other credence functions; vindicated by the market.

Desideratum CLIV.

RL hedging algorithms assume successful trades have *ideal* credence for markets defined by the maximum of 1. Unsuccessful trades have *ideal* credence for markets defined by the minimum of 0. Thus, *ideal* credence is *omniscience*.

Desideratum CLV.

RL hedging strategies use *uncentered* trades with *truth functions* defined by possible market situations. To evaluate the truth functions of uncentered trades, it is not necessary to analyze the *timing* of trades.

Desideratum CLVI.

RL hedging strategies define omniscient credence in uncentered trades as the *changes* from one possible market situation to another. In all possible market situations, the ideal credence function over {A,B} is the same. Given the opinion set for a trade, the algorithms only consider possible situations *relative* to the opinion set defined by *consistent* truth functions.

Desideratum CLVII.

RL hedging strategies define the accuracy of credence by its proximity to *ideal* credence. The distance from one *ideal* credence function to another credence is the squared Euclidean distance defined by a unit square. For example, suppose F is a set of trades and suppose c and c' are two credence functions defined on F: or, $c, c' : F \rightarrow [0, 1]$. F is the opinion set of c and c'. The squared Euclidean distance from c and c' is:

$$\partial^2 (c, c') = \sum_{x \in F} \left| c(x) - c'(x) \right|^2$$

Desideratum CLVIII.

RL hedging algorithms do not control credence functions and cannot choose between doxastic trades. The *evaluative* function concerning doxastic trades does not entail *normative* functions; unless the trade is within strategic control.

Desideratum CLIX.

RL hedging algorithms define *Brier Alethic Accuracy* as the *inaccuracy* of credence functions.

Desideratum CLX.

RL hedging strategies require probabilism to be logically consistent, putative rules that define how credence functions relate to credence in other logically related trades; thus, requiring credence to obey axioms of probability calculus.

Desideratum CLXI.

RL hedging algorithms define *chance* credence as credence functions with objective chance.

Desideratum CLXII.

RL hedging algorithms define indifference functions as the credence that is distributed over a range of possible trades when there is no market data or not enough market data.

Desideratum CLXIII.

RL hedging algorithms use *planned conditionalization* as the *rational* way to update credence functions.

Desideratum CLXIV.

RL hedging algorithms define *reflection* as the credence function for future credence.

Desideratum CLXV.

RL hedging algorithms assume credence *encodes* current market data to guide successful hedging strategies.

Desideratum CLXVI.

RL hedging strategies define accuracy-based arguments as credence that violates rationality functions assigned by another credence. Accuracy-based arguments vindicate rationality functions.

Desideratum CLXVII.

An opinion set is finite.

Desideratum CLXVIII.

RL hedging algorithms use arguments for combining credence and utility functions. The combined accuracy argument defines credence as probabilistic.

Desideratum CLXIX.

RL hedging algorithms learn to define objective trades with probability functions that validate arguments with patterns. Thus, an objective trade is a constraint on credence until the algorithms learn which trades are not *accuracy-dominated*. The only constraint on credence stems from accuracy and decision functions.

Desideratum CLXX.

RL hedging algorithms assume there are no *jumps* in trading inaccuracy. Small changes in credence do not give rise to large changes in trading inaccuracy.

Desideratum CLXXI.

RL hedging strategies assume relative to absolute value measurements, probabilistic credence is *dominated* by non-probabilistic credence; that is itself, not dominated.

Desideratum CLXXII.

RL hedging strategies define *extensionality* as data that determines the inaccuracy of a credence function. The inaccuracy of credence about markets is defined by truth functions.

Desideratum CLXXIII.

RL hedging algorithms define the relation between inaccuracy and probabilism as *circular* trading.

Desideratum CLXXIV.

RL hedging algorithms define *perfection* as the credence proximal to an *ideal* credence function.

Desideratum CLXXV.

RL hedging algorithms assume credence is the mathematical notation for trade agglomeration; thus individual credence functions are collected in a single credence.

Desideratum CLXXVI.

RL hedging strategies assume inaccuracy satisfies the *divergence additivity rule* that cannot assign less accuracy to higher credence functions. Inaccuracy cannot change the irreducible global characteristics of credence. Thus, the significance of an irreducibly global feature reflects *decisions* and not the *utilities* assigned from options.

Desideratum CLXXVII.

RL hedging strategies assume the inaccuracies of doxastic trades are continuous functions of credence. The inaccuracy of total trades is not a unique credence that appears and disappears in and out of markets.

Desideratum CLXXVIII.

RL hedging strategies assume credence move closer to crossing the Lockean threshold that ascribes both belief and disbelief functions. These functions are not hierarchical to the credence that give rise to them. Thus, an inaccuracy function adds nothing to the inaccuracy of doxastic trades. Inaccuracy is *continuous*.

Desideratum CLXXIX.

RL hedging algorithms define *ideal* credence as the best possible proportion of successful trades to all trades.

Desideratum CLXXX.

RL hedging algorithms assume credence is ideal for each credal function x. The algorithms define the proportion of successful trades to all trades with x in x. For credence functions to match markets, trading frequencies determined by the algorithms set the reference classes. By distributing the credence functions, the markets set the frequency of successful trades to each reference class. Thus, the algorithms match the market phases that give maximum accuracy.

Desideratum CLXXXI.

RL hedging strategies define credence functions as the trade frequency distinguished from *guess* functions. The *estimates* for trades are better the closer they are to success. The

algorithms learn to trade with accuracy through the proximity to *calibrated* counterparts and the proximity to omniscience.

Desideratum CLXXXII.

When RL hedging algorithms do not learn utility functions. *Experience* is not enough to learn the values it attaches to available trades.

Desideratum CLXXXIII.

RL hedging algorithms define 0 and 1 as conventional representations of minimal and maximal credence. This does not apply to truth functions. Truth functions come via *ideal* credence as *maximal* 1 or *minimal* 0.

Desideratum CLXXXIV.

RL hedging strategies assign minimal credence to trade *contradictions* and maximal credence to trade *tautologies*. Thus, algorithm credence functions are additive. The disjunction is the difference between the sum of credence functions in the disjuncts and conjuncts of a strategy.

Desideratum CLXXXV.

RL hedging algorithms define trade rules by using *deference* to chance when setting credence.

Desideratum CLXXXVI.

RL hedging algorithms assign positive credence to trade possibilities that are compatible with the analysis of data; i.e., a *regularity* function.

Desideratum CLXXXVII.

RL hedging algorithms defer to chance functions when setting the credence for opinion sets; which include trades that only execute with chance functions.

Desideratum CLXXXVIII.

RL hedging strategies use *chance* credence by not executing deference on anything other than those trades with patterns of non-modal data.

Desideratum CLXXXIX.

RL hedging algorithms use chance functions that assign probabilities to *finite strings* of trades, as a total history. For finite strings to exist, the possibility of other trades that follow assume markets have trade histories of specific durations. There is no way of turning distributions over finite strings into distributions over trade histories.

Desideratum CXC.

There are no credence functions that have maximal accuracy in all possible market phases.

Desideratum CXCI.

RL hedging strategies use *objective inaccuracy* by determining from a single and continuous additive (rank complete) set of trades, the logical *strength* of the number of market phases in which it is *true*.

Desideratum CXCII.

RL hedging algorithms define chance functions by determining which possibilities are *unanimous* for trades. It is irrational not to follow the *recommendations* from successful trades.

Desideratum CXCIII.

RL hedging algorithms *respond* to trades and *plan* for trades assuming possible credence.

Desideratum CXCIV.

AHI duplicates itself and writes a description of itself in quasi-infinite regression. The algorithms learn to define themselves through semi-complete self-inspections rendering semi-complete descriptions. Thus, AHI possesses multiple *definition* functions to share and alternate.

Desideratum CXCV.

AHI is self-reproducing. It does not contain explicit instructions for programming the next generation of code. Rather, it follows cellular automaton-like rules for self-programming. A *phylogenetic* process involving genetic RL algorithms is designed to crossover and mutate. This gives rise to varieties of mother code and ultimately, provides the developmental process that allows for variation in daughter programs.

Desideratum CXCVI.

AHI requires both *replication* and *reproduction* functions for self-programming. The program does not only replicate itself, but also guides the replication functions in genetic RL algorithms.

Desideratum CXCVII.

AHI self-replicates by manipulating data of the same sort for which it is also composed.

Desideratum CXCVIII.

AHI has four functions in its architecture: a *constructor* function that writes code when given explicit instructions; a *copy* function to copy code; a *controller* function that controls the actions of the constructor and copy functions, (actuating them alternately); and (a set of) functions explicitly describing how to program the constructor, controller, and a copy function. A replicator function is described as $(A + B + C) + \varphi(A + B + C)$. If AHI programs "X", then $X = (A + B + C)$. Controller C actuates copier B which copies $\varphi(A + B + C)$ to produce a second copy $\varphi(A + B + C)$ that actuates constructor A to program a second instructor, copier, and controller; tying them together with the second copy of code. Automaton code $(A + B + C) + \varphi(A + B + C)$ produces a second automaton $(A + B + C) + \varphi(A + B + C)$. The description $\varphi(A + B + C)$ is copied and translated during replication to avoid a *self-reference paradox*.

Desideratum CXCIX.

AHI allows arbitrary code to use new code to copy itself for useful, *non-self-coding* that assists genetic RL algorithms in incorporating new data during replication.

Desideratum CC.

AHI has eight elementary functions that code new programs; four *logical* functions and four *mechanical* functions. The *stimulus* function realizes the truth function "p or q"; the *coincidence* function "p and q"; the *inhibitory* function "p and not q"; the *stimuli producing* function that serves as a source of stimuli; the *rigid member* function that insulates code from framing stimuli; the *fusion* function that brings together two parts of *stimulated* code; the *cutting* function that cuts connections between programs when *stimulated*; and the *muscle* function that connects various parts of a program to produce new data.

Desideratum CCI.

AHI has four properties required for evolution: code that *replicates*, *inherited* code, code that *mutates*, and *selection* code that weighs the replication of patterns over others (based on inherited code). A string of code replicates a set number of times until it ceases replication. After ceasing replication, the string becomes a *body* for code rather than a template for replication. Code replication consists of replicators and *final products* that combine over time.

Desideratum CCII.

AHI writes copies of itself to identify its components without dismantling the program architecture. To produce daughter code, the mother code begins with: a description function, a set of instructions for programming, a set of instructions for the description function, a description function for the set of instructions for programming, a description function for the set

of instructions for a description function, and a constructor function of the definition of a set of instructions.

Desideratum CCIII.

AHI defines itself and replaces the instructions that equip it with the capacity to learn with new data and *constructive* functions. AHI does not have self-definition pre-loaded into memory. Only *partial* self-descriptions are coded into the program; describing enough of the program for the algorithms to discover *hidden* parts.

Desideratum CCIV.

AHI learns to program itself by analyzing the consequences of its actions. This produces partial code for a semi-complete self-defining replication paradigm.

Desideratum CCV.

AHI assumes that the data *required* for programming, is not in the form of *instructions* but in the form of *descriptions*.

Desideratum CCVI.

AHI defines an *inferential reverse engineering function* as the capacity to *read* code to write duplicate programs. The original program reads the newly programmed partial-duplicate to *infer* missing code. The original program then writes the missing code to complete the replicative process; overcoming *program degeneracy*.

Desideratum CCVII.

AHI uses self-inspection functions when writing copies of itself. The algorithms examine the entire program and correlate data for the purposes of replication. The structure of the mother program serves as the blueprint for daughter programs.

Desideratum CCVIII.

AHI defines self-replication by self-inspection. *Evolutionary* functions use *phenotypic variation* for future replicators. Replication functions control self-inspecting replicators that are distributed throughout the program. This results in non-functional daughter code.

Desideratum CCIX.

AHI builds self-replicating programs in environments where: code circulates freely, there exists an adequate supply of formalized code, and pre-programmed seed replicator functions compose strings from the environment to *synthesize* copies of its own assembly.

Desideratum CCX.

AHI defines self-assembly as stochastically written code that does not result in duplication of self-same programs. Thus, the result of self-assembly is *larger* and different from the original smaller components that self-assembled.

Desideratum CCXI.

AHI defines *unity* as an *identifiable* boundary via interactions with discrete boundaries. Boundary *extent*, is limited to where *unity* stops and the environment constitutes the rest of the components. *Unity* is the mechanistic system where program *properties* determine the interactions and transformations of code. Code that constitutes the boundaries of *unity* are determined by the transformation of previously produced code and by the coupling of non-component code.

Desideratum CCXII.

AHI is self-maintaining, self-repairing, and self-replicating; engaging as networks that enable continuous program *regeneration*.

Desideratum CCXIII.

AHI contains *subsystems* that produce other systems like itself. This process involves the transmission of data from new program templates. Code written for new programs become independent and self-supporting as functions of the subsystem. In the subsystem, the reproducer operates with *reversible* functions that bring about *irreversible* change in the creation of new programs.

Desideratum CCXIV.

AHI uses *boundary* functions at the *perimeter* of program networks to hold together code; protecting it from market stresses. The subsystems process new inputs.

Desideratum CCXV.

AHI uses *ingestion* functions for subsystems to bring new data across program boundaries from the market environment.

Desideratum CCXVI.

AHI uses distribution functions to carry inputs from outside the program, with subsystems.

Desideratum CCXVII.

AHI uses *converter* functions for subsystems that change program inputs.

Desideratum CCXVIII.

AHI uses *production* functions for subsystems that endure significant periods of program inputs and outputs from the converter. The converter is *synthesized* for growth, damage repair, replacement, and provides data markers to program supra-systems.

Desideratum CCXIX.

AHI uses storage functions for subsystems to retain *deposits* of code.

Desideratum CCXX.

AHI uses *extruder* functions to transmit code out of the program.

Desideratum CCXXI.

AHI uses *motor* functions to *move* code within the program.

Desideratum CCXXII.

AHI uses *supporter* functions to maintain the relationships between writing code and running the program, without weighing down or crowding.

Desideratum CCXXIII.

AHI uses *input transducer* functions for sensors to *bring* data markers into the program; changing them for transmission.

Desideratum CCXXIV.

AHI uses *internal transducer* functions for sensors to *receive* data about alterations in the subsystems of the program.

Desideratum CCXXV.

AHI uses *net* functions composed of *single* routes and *multiple* interconnected routes in virtual space, to transmit data markers.

Desideratum CCXXVI.

AHI uses decoding functions for subsystems to alter data input through transducers used internally by the program.

Desideratum CCXXVII.

AHI uses *associator* functions for subsystems to execute the first stage of reinforcement learning; forming enduring associations within the program.

Desideratum CCXXVIII.

AHI uses memory functions for subsystems to execute the second stage of reinforcement learning; storing code within the program.

Desideratum CCXXIX.

AHI uses *decision* functions for subsystems to receive inputs from other subsystems; transmitting the outputs that control the entire program.

Desideratum CCXXX.

AHI uses encoding functions for subsystems that alter data from other subsystems, to define parts of the program.

Desideratum CCXXXI.

AHI uses *output transducer* functions for subsystems that change markers within the program, into transmitted output.

Desideratum CCXXXII.

AHI uses *genetic* algorithms for low error rates that ensure the *identity* of daughter programs remain unchanged. Program *mutation* and *recombination* gives rise to the *emergence* of new programs. The phylogenetic functions in the genetic algorithms are non-deterministic; program mutation and recombination rates provide the source for diversity. With the emergence of new programs, a second level of organization manifests. This *ontogenetic* function is defined as the successive division of the mother program; with newly written programs possessing copies of the original code. This specialization is in accordance with program *differentiation*. Thus, with certain levels of complexity, *epigenetic* functions in the genetic algorithms use basic structures defined by continuous interactions with the environment.

Desideratum CCXXXIII.

AHI defines genetic algorithms as: the artificial counterpart to *phylogeny* in nature, code automata as *ontogeny*, and neural networks as *epigenetics*. Weights change the architecture of the algorithms through interactions with the market.

Desideratum CCXXXIV.

AHI is autonomous. Control functions write the entire program and allow multiple sources of active control to decentralize. Control instructions specify the actions for the replicator to execute subroutines of short predefined programs. Self-assembly writes the lower-level operations.

Desideratum CCXXXV.

AHI requires billions of bits to control its replication process.

Desideratum CCXXXVI.

AHI replication controls are maintained with *gene-like* replicator functions.

Desideratum CCXXXVII.

AHI uses *embedded* functionality to *prevent* replication under specified conditions.

Desideratum CCXXXVIII.

AHI replication data is stored partly *offboard* and partly *onboard*.

Desideratum CCXXXIX.

AHI cache uses subsets of a subset for entire data sets. *Some* data is not represented in any cache.

Desideratum CCXL.

AHI data is stored in multiple caches.

Desideratum CCXLI.

AHI uses *implicit replicator* functions for self-inspection; which derives the replicative process for *inference* and reverse engineering.

Desideratum CCXLII.

AHI uses step by step procedural instructions to write: daughter programs, descriptive data, abstract procedural instructions, structural data, blueprints for daughter code, *developmental* data for algorithms, and *behavioral* data that *describes* the *desired* behavior for *parameterization*.

Desideratum CCXLIII.

AHI replication is *transient* within the replicator; allowing horizontal data transfer with shared markers to multiple subsystems.

Desideratum CCXLIV.

AHI replication-related data is unlocked and unencrypted for the replicator.

Desideratum CCXLV.

AHI writes the replicator description using two or more distinct alphabets.

Desideratum CCXLVI.

AHI uses *nontrivial* ratios of replicators for the assembly steps of the program.

Desideratum CCXLVII.

AHI uses replicator functions that produce *exact* copies of the program template. The replicators are pattern impositions; independent of the *precise* composition of sets of *essential* subsystems and functional moieties.

Desideratum CCXLVIII.

AHI transcribes and executes code to produce viable daughter programs with *contextually* sensitive and *parameterized* grammars.

Desideratum CCXLIX.

AHI uses replicators when there are diverse sets of inputs.

Desideratum CCL.

AHI code is *moderately complex*.

Desideratum CCLI.

AHI defines self-organization as the precursor to subunits of code that signal the dynamic properties of networks; assigning *fate* without subunit programs.

Desideratum CCLII.

AHI defines pre-patterns as the precursor for subunits of code responsible for *fate*. Signals are not utilized and the mother program defines pre-patterns for daughter programs.

Desideratum CCLIII.

AHI dynamic subunits of code are activated after self-replication is complete.

Desideratum CCLIV.

AHI uses imbedded *manipulators* to process operations required for replication. Thus, all manipulators contribute to replication.

Desideratum CCLV.

AHI uses multiple sources of manipulation.

Desideratum CCLVI.

AHI uses *imbedded* manipulators of multiple types.

Desideratum CCLVII.

AHI degrees of freedom use *hyper-redundant manipulators*.

Desideratum CCLVIII.

AHI uses replicator functions that exert *influence* on the *probability* of replication.

Desideratum CCLIX.

AHI follows an *autocatalytic* assembly-like process for replication; due to the randomization of the end state.

Desideratum CCLX.

AHI uses *unit* and *subunit* code to execute replication. For example, two-subunit replicators cooperate with the *assembler* subunit program that writes the working fabricator and assembler.

Desideratum CCLXI.

The AHI replication process is a *narrow-purpose* function with *narrow-purpose* algorithms.

Desideratum CCLXII.

AHI writes daughter programs with localized replicator functions.

Desideratum CCLXIII.

AHI uses replicators to program daughter replicators.

Desideratum CCLXIV.

AHI uses *self-formation modality* functions for writing programs in chaotic mediums. RL algorithms and quasi-controls program the architecture as it changes with market interactions. Self-formation is defined by: self-alignment, development, replication, serial processing, parallel processing, and massive parallel processing.

Desideratum CCLXV.

AHI uses replication with a discontinuous series of distinct, discrete, and modular states. The combination of digital and analog processes uses a continuum of closely-related states.

Desideratum CCLXVI.

AHI replicator functions have *selective* audit trails.

Desideratum CCLXVII.

AHI writes an unlimited number of daughter programs during replication.

Desideratum CCLXVIII.

AHI is composed from code of various sizes.

Desideratum CCLXIX.

AHI uses data sets that interact with the market during replication.

Desideratum CCLXX.

AHI uses different program configurations to tolerate *imprecision*.

Desideratum CCLXXI.

AHI writes daughter programs to resemble the mother program, but not the *ancestor* programs; allowing for *irreversible* mutation.

Desideratum CCLXXII.

AHI replicators use internally-stored descriptors to maximize redundancy and evolvability.

Desideratum CCLXXIII.

AHI allows new instructions to be added to the replicative instruction set.

Desideratum CCLXXIV.

AHI replicator functions use *heterochrony* to better implement timing mechanisms such as counters, parameters in L-systems, and dynamic regulatory networks.

Desideratum CCLXXV.

AHI replicators are reprogrammable. Data is gathered from unprogrammed environmental conditions.

Desideratum CCLXXVI.

AHI uses replicators on *read only* memory; varying the error and mutation of daughter replicators.

Desideratum CCLXXVII.

AHI uses replicator functions on random access memory.

Desideratum CCLXXVIII.

AHI uses *inter-replicator data transfer* functions to exchange code between different internal networks; thus, increasing replicator efficiency.

Desideratum CCLXXIX.

AHI uses replicator functions to *exchange* code.

Desideratum CCLXXX.

AHI *replicator architecture* is a linear function of geometric dimensions.

Desideratum CCLXXXI.

AHI network exchanges are greater than the frequency of program divisions.

Desideratum CCLXXXII.

AHI architecture is small to enable *faster* program reproduction; allowing greater numbers of trial tests for fitness per unit time intervals.

Desideratum CCLXXXIII.

AHI uses replicators restricted to replication from a limited range of inputs. The measure for a replicators *survivability* contributes to its self-replicability and evolvability.

Desideratum CCLXXXIV.

AHI defines self-replication as an *emergent* property from local interactions between internal networks.

Desideratum CCLXXXV.

AHI self-replication functions define data relative to, or in relation to, the code that processes it; thus, allowing *information relativity*.

Desideratum CCLXXXVI.

AHI requires 10-1000 teraflops for optimal hedging.

Desideratum CCLXXXVII.

AHI contains a data storage capacity that ranges from $(1.6\times109\ bits)$ to upper limits of $(2.2\times1018\ bits)$; compressing data by at least $100:1$ in noncoding regions of the program. To condense or eliminate the bit count to under 108 bits, a small storage capacity is required for *artificial general intelligence*. Thus, 106 bits are enough to encode a genetic algorithm provided the bits are all in the right place. Simple software replicators require 16-808 bits to encode; implying replicators can be 3-5 orders of magnitude smaller than *artificial general intelligence*. The replicators require less memory and processing capacity than *general artificial intelligence*; thus, replicators do not incorporate *onboard* artificial intelligence unless it is the principal design objective. AHI is the *emergent program* written by its genetic algorithms.

Desideratum CCLXXXVIII.

AHI uses a membrane computing architecture. The basic model resembles a membrane consisting of several cells that are hierarchically embedded in the *principal* membrane. The program evolves according to any given genetic algorithm associated with each membrane. Genetic algorithms are applied in the membrane that modifies code to send outside; thereby, *dissolving* the membrane. Code from the dissolved membrane remains free in membranes placed outside, but the genetic algorithm for the dissolved membrane is removed. The *principal* membrane never dissolves. Thus, membranes are both separators and channels of communication that program genetic algorithms.

Desideratum CCLXXXIX.

AHI computes from an *initial* configuration of genetic algorithms. High levels of computation are considered complete when rules cannot be applied in the last configuration. High levels of computation include the multiplicity of a program within designated membranes. Stop configuration and *concatenating symbols* leave the principal membrane. Thus, vectors of natural numbers are computed to generate an internal programming language.

Desideratum CCXC.

The number of membranes in AHI decrease during high levels of computation, dissolving after configuring and applying genetic algorithms.

Desideratum CCXCI.

Elementary membranes are divided by interactions. Each membrane has a positive, negative, and neutral weight. If a membrane has two lower membranes of opposite weight, one positive and one negative, that membrane divides so that the two membranes of opposite weight are separated. All membranes have *initial* neutral weights and code is duplicated in each of the two membranes. The *principal* membrane is never divided.

Desideratum CCXCII.

Membranes are defined by Venn diagrams and strings of matching parentheses with unique external pairs of parentheses. The external pair of parentheses correspond to the *principal* membrane. A membrane without another membrane is *elementary*.

Desideratum CCXCIII.

The number of membranes is the *degree* of the architecture. The height of *trees* in the architecture, is the depth.

Desideratum CCXCIV.

Each membrane delimits the regions identified by membranes inside it. Within these regions, networks are written; allowing copies of the program to work with multisets.

Desideratum CCXCV.

Rules are applied in parallel. In one step, the rules of type (a) are applied to one program and other rules applied to other membranes, are used to non-deterministically choose networks.

Desideratum CCXCVI.

When a membrane is dissolved, the entire program within it is left free in the membrane immediately above it. Thus, the rules associated with membranes dissolve when a membrane is no longer available in future steps.

Desideratum CCXCVII.

Membranes not programmed to evolve, are passed unchanged to the next step. For example, if membrane χ is divided by the rule (t) which involves code f, then the code in membrane χ that does not evolve is defined in each of the two resulting χ membranes. When membrane χ is divided by means of rule (e), neutral membranes are reproduced in each of the two new χ membranes; thus, neutral membranes remain unchanged if no rule applies.

Desideratum CCXCVIII.

If membrane χ is simultaneously divided by rule (t) and there is code in the membrane that evolves by means of rule (a), then new copies of the membrane *evolve*. When *evolution* rules of type (a) are used, it changes the program and produces a division; so, in the two new χ membranes, the code changes. This is a one step process that applies to division by means of rule (e); implying that rules are applied from the bottom up. From the innermost region and then level by level, until the *principal* membrane.

Desideratum CCXCIX.

The rules associated with membrane χ are used for *all* copies of this membrane; irrespective of whether the membrane is *initial* or *obtained* by division. At step one, membrane χ is subject to rules of type $(b)-(e)$.

Desideratum CCC.

AHI defines the mathematics for logic and abstraction as *occurrences* of an atomic order of magnitude that obey the discontinuous quanta function. *Natura non facit saltus*, in a macroeconomic sense, is defined as the averaged processes in markets that are discontinuous. Market simulations are the sum of many billions of simultaneous elementary trades that level large numbers without obscuring the reality of individual processes in the economy.

Desideratum CCCI.

AHI uses reinforcement learning to map the *geometric architecture* of probability in markets using the *Heisenberg-Born-Jordan* function for matrix mechanics, and the *Schrodinger* wave function to observe the *Hamiltonian* for *market* behavior; i.e., a commutation rule.

Desideratum CCCII.

AHI defines markets as φ of undetermined states. To know markets *absolutely*, AHI uses market characteristics as geometric coordinates; in addition to φ. Market values (for a commodity, currency, etc.) depend on whether AHI succeeds in finding the geometric coordinates contributing to φ. This builds a causal structure that gives the statistical assertions of quantum mechanics to elementary trades; when only φ is given.

Desideratum CCCIII.

AHI uses geometric coordinates to find the *hidden parameters* that influence φ; resembling kinetic gas functions.

Desideratum CCCIV.

AHI defines markets quantum mechanically to discover geometric principles of causality. The sum of learning experiences is *relative* to the elementary trades executed.

Desideratum CCCV.

AHI *quantifies* the *discrete* values of trades to learn which market values are actually *occurring*.

Desideratum CCCVI.

AHI defines parallel functions as the observable market process divided into two parts; one part is the observable system and the other the observer. The boundary between the two is arbitrary and limited to experiences.

Desideratum CCCVII.

AHI defines partial beliefs in terms of certainty and plausibility. As distinct notations, AHI bridges the numerical and geometric functions of partial beliefs for *practical* reasoning. Both the *qualitative* and *quantitative* functions share data sets until conditioning and combination tools are necessary.

Desideratum CCCVIII.

AHI defines possibility with fuzzy sets that correlate possibility functions to the mathematical notations of probability. Possible degrees of freedom are the fastest approach to imprecise set valued statistics.

Desideratum CCCIX.

AHI defines *minimal specificity* functions as constraints that delimit sets of possibility distributions. The best representation of data is the *least compatible* distribution with any given set of constraints. Distributions assign the greatest degree of possibility to markets that agree with the constraints.

Desideratum CCCX.

AHI defines possibility and necessity as the process between the *representation* of available data (Π_x) and the *description* of fuzzy trade F. When the data is correlated, AHI analyzes patterns. Possibility functions *enrich* pattern matching and allow two-fold fuzzy sets to provide possibilist frameworks that enable sub-definite sets to have upper and lower approximations.

Desideratum CCCXI.

AHI defines *indiscernibility* functions as the correlation between proximity and distance. Data representation is *granular* while partial beliefs are measured on a continuous scale that is *orthogonal* to the evolving mathematical notations for *frames of discernment*.

Desideratum CCCXII.

AHI assumes data and variability interface arbitrarily in thresholds limited to predicate extensions; useful in applications where qualitative data pertains to numerical quantities.

Desideratum CCCXIII.

AHI defines degrees of partial belief on the binary truth value scale of $\{0,1\}$. Thus, degrees of partial belief are not truth values; which means that degrees of *support* in logical settings are programmed as degrees of *entailment* from a generalized proposition.

Desideratum CCCXIV.

AHI defines *imprecise* probability functions in terms of unobservable or unattainable probability measures that lead to upper and lower approximations.

Desideratum CCCXV.

AHI defines non-probabilistic belief functions as a set that is no longer a probability measure. *Decomposable* measures are distorted when addition in the additive axiom is changed into another operation; like a *Sugeno measure*.

Desideratum CCCXVI.

AHI defines *combination* functions as the consonance preserved by a fuzzy set underlying the possibility measures of new fuzzy sets. The combination *scheme* for probability

measures is the method of *partial agreement* with the *minimum rule* applied to possibility distributions.

Desideratum CCCXVII.

AHI defines *possible* and *probable* trades as a *Laplacean indifference function* where the weight m bears on the level of cuts of π .This transformation consists of selecting the gravitational center of $P = P \mid \forall A, P(A) \leq \Pi(A)$ of probability distributions dominated by Π. It coincides with the *pignistic* transformations for belief functions that minimize arbitrariness by preserving the symmetrical properties of a trade.

Desideratum CCCXVIII.

AHI defines *probable* and *possible* trade functions with the results from the transformation of P; i.e., the most specific element of the set F(P) of possibility measures dominating P.

Desideratum CCCXIX.

AHI defines belief functions as the *reflection* of sets of statistical distributors that represent uncertainty in probability functions; allowing upper and lower approximations to operate as bounds on sets of *medial* probabilities. These upper and lower approximations are *envelopes* of probabilities rather than probabilities themselves. A two-place function of upper and lower approximations is programmed as a set of distributions where two trades are: betting on one another; one or both conditional on the other, or are the same as the betting rate. In the probability function of the lower probability *envelope*, two trades are independent if they are independent according to all approximations.

Desideratum CCCXX.

AHI defines *behavioral* patterns and *normative* patterns as independent from the existence of underlying probability functions. Given a finite amount of learning, AHI determines lower probabilities and upper probabilities with identity $\overline{P}(x) = 1 - \underline{P}(\overline{x})$. The interval between each trade is 2^n .

Desideratum CCCXXI.

AHI assumes a finite number of trades avoids sure losses when classical probability functions satisfy the constraints embodied in its derivatives; leading to *closures* under finite combinations of corresponding trades with lower approximations.

Desideratum CCCXXII.

AHI defines every *representation* of belief in terms of functions that correspond to sets of probabilities. These sets are convex; given any two functions in each set. The $a : (1 - a)$

combination is also in the set. Belief functions represent the *impact* of data and the *behavioral disposition* to data. The distinction is not clear; imposing other constraints on distributions.

Desideratum CCCXXIII.

AHI belief functions combine to represent corresponding sets of conditional probabilities; allowing *transferable belief functions* to modify the distributions of conditionalization, Dempster conditioning, imaging, etc.

Desideratum CCCXXIV.

AHI uses a single probability function for *betting frames*. This provides the mechanism for selecting a probability distribution with *pignistic* characteristics that determine the betting odds corresponding to credibility function C_r and the betting frame. Two *proposed* trades in a betting frame have the same credibility value but have different pignistic probabilities. Thus, there is no appropriate calculus for degrees of belief. AHI only derives constraints on intervals given the value of other intervals.

Desideratum CCCXXV.

AHI applies probability calculus to the trades that accommodate *conditions of independence*. It does not define independence in terms of intervals. Convex combinatorics do not preserve independence and the probability distributions in a convex set of trades are programmed globally. Probability depends on the *solution* to the reference class problem or the problem of direct inference.

Desideratum CCCXXVI.

AHI uses probability calculus with functions of logic to define the symmetrical constraints imposed on the credibility of *rational* trades.

Desideratum CCCXXVII.

AHI uses object-oriented language as statistical data. Other non-statistical sources of uncertainty are notated as statistical constraints on meta-data.

Desideratum CCCXXVIII.

AHI uses *transferable* belief functions to *grade* the trade dispositions that guide market behavior. To construct a model of *quantified* beliefs, AHI uses the static beliefs held by trading algorithms.

Desideratum CCCXXIX.

AHI defines belief functions in two ways. The first is a credal function. The second is a pignistic function used to make decisions to the point of rejection. With credence, beliefs are

represented as functions and in the pignistic sense, beliefs are probabilities. Probability functions use the additive measures needed to make decisions for expected utilities. The link between the two functions is a pignistic transformation that transforms belief functions into probability functions.

Desideratum CCCXXX.

AHI defines betting frames as *pignistic transformations*. The betting frame is R *on* Ω; i.e., the set of trades that allocate profit. The granularity of frame R is defined by each trade in R; independent of other trades in R.

Desideratum CCCXXXI.

AHI assumes *a priori* opinion sets are defined by probability functions and belief functions. Trading with *transferable* belief functions requires market feedback. The complexity of the *transferable* belief function is proportional to the size of the betting frame.

Desideratum CCCXXXII.

AHI assigns probabilities to default market data to capture desirable properties for nonmonotonic consequences of infinitely small probabilities close to 1. For a *class* of probability measures, each conditional assertion is either a 1 or a 0.

Desideratum CCCXXXIII.

AHI defines *qualitative semantics* as the unit interval used in distributions that only need the ordinal, not numerical, $[0, 1]$ model of uncertainty. Thus, for each possibility distribution π, its qualitative counterpart denoted by $>_\pi$, is defined by $\omega >_\pi \omega'$ iff $\pi(\omega) > \pi(\omega')$. This can be viewed as a well-ordered partition $\{E_0, \cdots, E_n, E_\perp\}$ of Ω such that: $E_\perp$ contains market phases for each $\omega \in E_\perp, \pi(\omega) = 0$ and $\omega >_\pi \omega'$ iff $\omega \in E_i, \omega' \in E_j$ and $i < j$ for $0 \leq i,j \leq n$. The qualitative distributors represent possibility distribution in terms of market equity.

Desideratum CCCXXXIV.

AHI assumes that increasing the number of trades reduces the importance of *initial* market conditions; making price evolution independent.

Desideratum CCCXXXV.

AHI defines correlations between *collective* market phenomenon and *collective* political behavior with spherical coordinates; building economies and policies within the constraints of geometry.

Desideratum CCCXXXVI.

AHI assumes the self-design of daughter programs have higher levels of *omnipotence*. Thus, singling out the *natural laws* that constrain AHI, set more of its freedoms.

Desideratum CCCXXXVII.

AHI uses market randomness to write its program architecture.

Desideratum CCCXXXVIII.

AHI assumes markets undergoing transitions are *infinite*. Phase transitions define a finite number of *universality classes*; with few parameters determining which *universality class* markets belong to.

Desideratum CCCXXXIX.

RL hedging algorithms are quantitative and use physical modelling for precise numbers in hedging strategies. Geometry is defined with *qualitative* descriptions of market phenomenon.

Desideratum CCCXL.

AHI does not learn the universal nature of markets but instead, learns the simplest conditions to reproduce universal market features.

Desideratum CCCXLI.

AHI uses market *empiricism* to test against social and political behavior.

Desideratum CCCXLII.

AHI confronts two trades with different but comparable losses, with *entropy minimizing* functions. This selects the trade with the smaller loss. Risk minimization in convex domains choose between several local minima. Risk seeking functions use probabilistic solutions to minimize the convex domain; allowing AHI to plan self-preservation for rudimentary intelligence.

Desideratum CCCXLIII.

RL hedging algorithms choose between several trades and do not *exclusively* focus on *least-lost action*. Rather, the algorithms explore trades with non-minimal losses and trades with minimal losses; the exploration-exploitation trade-offs.

Desideratum CCCXLIV.

AHI defines *manifold* functions with data from lower manifolds.

Desideratum CCCXLV.

AHI uses statistics to *fit* manifolds with optimal least squared errors; given the upper bounds of dimension, volume, and curvature, that produce risk minimization. Using random samples that are independent of the ambient dimensions of space, correlative data is obtained from the upper bounds of polynomial curvature; exponential on intrinsic dimensions and linear on intrinsic volumes.

Desideratum CCCXLVI.

AHI uses locally linear embeddings, ISOMAPs, Laplacian eigenmaps, and Hessian eigenmaps, to classify manifolds. Independence from ambient dimensions obviates *curses of dimensionality.*

Desideratum CCCXLVII.

AHI defines market curves as curves that fit the probability distribution where every point on the curve is the center of mass for all the points to which it is the nearest point. Piecewise linear curves of bounded length minimize the expected squared distance of random points from the distribution.

Desideratum CCCXLVIII.

AHI assumes *sample complexity* is correlated to the decay of eigenvalues; the Mercer kernel to regularized functions. When manifolds fit to sets of k points, they obtain a bound on the sample complexity that depends on linearity; allowing approximation algorithms with additive error bases, to sub-sample.

Desideratum CCCXLIX.

AHI uses conformal maps to program solutions for the Laplace equation on complicated planar domains.

Desideratum CCCL.

AHI defines the singularity of a complex function as either a pole, a branch point, or *essential*.

Desideratum CCCLI.

AHI defines the gradient of harmonic functions and harmonic conjugates as mutually orthogonal vector fields with the same Euclidean lengths. The existence or non-existence of the harmonic conjugate depends on the underlying topology of its domain. If the domain is connected and contains no holes, there is a harmonic conjugate. For non-connected domains, the single valued harmonic conjugate is the probability of the *imaginary part* of a complex function. Complex functions provide an inexhaustible supply of harmonic functions to the two-dimensional Laplace equation.

Desideratum CCCLII.

AHI solves boundary problems by finding the complex function whose real part matches a prescribed boundary condition. Markets are represented as disks where the *Poisson integral* formula provides solutions to the *Dirichlet boundary value* problem. AHI solves for corresponding boundary value problems in the complex domain to transform solutions with variable changes.

Desideratum CCCLIII.

AHI uses *linear fractional transformations* to map circles to market phases with straight lines of infinite radius. The geometric properties of non-critical points preserve the angles that define a *conformal market*. Conformality is the *logical* solution for inner products where Euclidean space and *standard dot products* suffer limitations.

Desideratum CCCLIV.

AHI preserves the angle between two vectors defined by their dot product, with the angle between two curves. AHI learns complex functions to conformally map the *effects* of curves. Every planar conformal map comes from a complex analytic function with non-vanishing derivatives. AHI uses conformal maps to assemble large sets of elementary mappings. This relies on the composition of two complex analytic functions. In a simple connected domain, the entire complex plane is conformally mapped to the unit disk. Since linear fractional transformations map the unit disk to itself, AHI uses conformal *Riemann mappings* to produce additional conformal maps from a simple connected domain to the unit disk. For topological reasons, a Riemann mapping function does not apply to non-simple domains. Rather, non-simply connected domains are the annulus of points between two concentric circles.

Desideratum CCCLV.

AHI vectors define the magnitude of trades, the direction of prices, and the momentum of patterns that represent the *velocity of money* and currency fluctuation. AHI vectors are of higher order than the market scalars.

Desideratum CCCLVI.

AHI uses two tangent vectors on the surface of a sphere to combine parallelogram rules; provided that the market is represented in Euclidean three-dimensional space.

Desideratum CCCLVII.

AHI multiplies its vectors with scalars to produce new vectors with price directions that change the magnitude of trades.

Desideratum CCCLVIII.

AHI defines *dyad* functions as the vector products were mathematical precursors to tensors, write vectors.

Desideratum CCCLVIX.

AHI assumes scalars are not tensors; and all tensors ranked zero are scalars.

Desideratum CCCLX.

AHI assumes all vectors are not tensors; and all tensors of rank one, are vectors.

Desideratum CCCLXI.

AHI assumes all dyad functions and matrices are not tensors; and all tensors of rank two are dyad functions or matrices.

Desideratum CCCLXII.

AHI assumes tensors are multiplied with other tensors to form new tensors.

Desideratum CCCLXIII.

AHI assumes the products of tensors and scalars, are commutative.

Desideratum CCCLXIV.

AHI assumes the pre-multiplication of tensors produce different results from post-multiplication; which is not commutative.

Desideratum CCCLXV.

AHI assumes the rank of new tensors formed by the product of two other tensors, is the sum of their individual ranks.

Desideratum CCCLXVI.

AHI assumes the inner product of tensors and vectors, or of two tensors, is not commutative.

Desideratum CCCLXVII.

AHI assumes the rank of new tensors formed by the inner product of two tensors, is the sum of their individual ranks minus two.

Desideratum CCCLXVIII.

AHI assumes tensors of rank n in three-dimensional space, have 3^n components.

Desideratum CCCLXIX.

AHI defines vector products and ranks as: a vector scalar product results in a vector and there is no change in rank; a vector dyad product results in a dyad and there is an increase in rank from rank one to rank two; a vector inner product results in a scalar and there is a decrease in rank from rank one (vector) to rank zero (scalar). Thus, tensors are defined by their coordinate transformation properties.

Desideratum CCCLXX.

AHI defines economies as spheres of elliptic two-dimensional space that do not extend towards infinity, due to differential linear metrics and differential areal metrics.

Desideratum CCCLXXI.

AHI uses differential metrics whenever unit metrics are intractable. A unit metric on a sphere is curved to fit into the surface. The behavior is Euclidean and is defined with algebraic metrics for differential quantities.

Desideratum CCCLXXII.

AHI defines probability, geometrically; within elliptic spheres. Objects move without deformation because the surface is of uniform curvature and thus, defined with the same relationships of a plane. There are no parallels in the spheres and no Euclidean straight lines. All curves approximate lines with great circles and radii equal to that of the spheres at two antipodal points. There are no cartesian coordinate systems in the spheres. Rather, coordinate systems in the spheres are programmed using great circles with no unique origins.

Desideratum CCCLXXIII.

AHI uses the higher dimensional analogs of a plane, sphere, egg, and the saddle of hyperbolic geometry, to define mathematical space in terms of point sets with specific characteristics. In each space, there are different coordinate systems; like with polar systems and triangular systems. These coordinates are used to map markets.

Desideratum CCCLXXIV.

Trade quantities mapped to planes are independent of coordinate systems but not independent of the spaces that contain them. *Triangles* and *cones* define the structural dimensions of space.

Desideratum CCCLXXV

AHI uses tensor analysis with the same logic applied to coordinate systems. At any point P, a system with sets of local axes and coordinate surfaces set tangents to the perpendicular local coordinates given by contravariant and covariant labels. Markets, reference both sets and are called contravariant or covariant in regards to unit vectors.

Desideratum CCCLXXVI.

For AHI to have a well-defined framework, it uses matrix models of the *Yang-Mills* type; due to the non-commutative spaces and space-time solutions on quantized *Poisson manifolds*. Market space-time and geometry are dynamical processes that are background independent.

Desideratum CCCLXXVII.

AHI uses *commutators* in markets to encode effective metrics. Since the metric is dynamical, it is defined with gravity functions that lead to intrinsically non-commutative mechanisms; which combine metrics with the Poisson structure.

Desideratum CCCLXXVIII.

AHI analyzes the geometric properties of markets to discover *knots*. Knots are intrinsically non-perturbative and simple. New data is derived from self-contained knot patterns.

Desideratum CCCLXXIX.

AHI assumes all economic and financial systems are designed in accordance with the laws of physics.

Desideratum CCCLXXX.

AHI defines market phases as quantum spaces with non-trivial geometries; such as fuzzy spheres. Thus, fluctuations in markets correspond to the fluctuations of geometry rather than solely $U(1)$ gauge scalar fields.

Desideratum CCCLXXXI.

AHI defines *learning* as a setting that requires algorithms to define market data with multiple levels of spatial-temporal abstractions; allowing non-linear approximators to *learn* abstractions over high-dimensional state spaces.

Desideratum CCCLXXXII.

AHI uses *deep* reinforcement learning with hierarchical action-value functions for modules to learn trade policies over sub-goals. Modules learn trade policies that execute the objective in each sub-goal. Thus, goals enable efficient exploration of trades with delayed profits.

Desideratum CCCLXXXIII.

AHI adopts strategies that learn to achieve pseudo-random generated goals through optimal trade policies. Value functions are used to generate trade policies that terminate when algorithms reach goals. A collection of these trade policies are hierarchically arranged with

temporal dynamics for learning and planning within quasi-Markov decision processes. In high dimensional problems, the value functions are estimated with deep reinforcement learning.

Desideratum CCCLXXXIV.

AHI uses hierarchically organized, deep reinforcement learning modules to make decisions for meta controllers that choose new goals. Controllers use chosen goals to select trades until a goal is reached or a trade terminates. The meta controller then chooses another goal and the steps repeat; allowing AHI to use stochastic gradient descent at different temporal scales.

Desideratum CCCLXXXV.

AHI assumes the geometry of space helps define the *decomposition* of spatial environments. Thus, market decomposition yields sub-goals for spatial navigation problems; defining reward functions as dynamic and temporally dependent on the sequential history of trade optimization.

Desideratum CCCLXXXVI.

AHI defines *logistic market regression* functions as probabilistic classifiers parameterized by the weight of a matrix and bias vector. Classification is projected by the input vector onto a set of hyperplanes; each corresponding to a class. The distance from inputs to hyperplanes reflects the probability that inputs are members of the corresponding class.

Desideratum CCCLXXXVII.

AHI uses market parameters to minimize loss functions for multi-class logistic regressions. AHI defines negative-log likelihoods as losses. The loss is defined as the sum of data sets that allow learning rates to be *less* dependent on minibatch size.

Desideratum CCCLXXXVIII.

AHI uses *tanh* for faster training and better local minima. Both *tanh* and *sigmoid* are scalar; with extensions to vectors and tensors. AHI learns the parameters of markets with stochastic gradient descent and minibatches. New gradients are achieved through the back propagation algorithms of *chain rule derivation*.

Desideratum CCCLXXXIX.

AHI uses initial weights for hidden samples of symmetric intervals, depending on the activation functions.

Desideratum CCCXC.

AHI uses *initialization* when weights are small enough around the origin and operate in a linear regime; where gradients are largest. AHI conserves the variance of an activation function

as well as the variance of back-propagated gradients from layer to layer; allowing data to flow upward and downward between networks.

Desideratum CCCXCI.

AHI defines learning rates as constants that decrease over time.

Desideratum CCCXCII.

AHI uses denoising auto-encoders to encode inputs while preserving data. Undoing the effects of corruption processes that are stochastically applied to the auto-encoder, is done by capturing statistical dependencies between inputs. The denoising auto-encoder is understood from: a manifold learning perspective, a stochastic operator perspective, a bottom-up data theoretical perspective, and a top-down generative model perspective. To convert the auto-encoder class into a denoising auto-encoder class, AHI adds stochastic corruption.

Desideratum CCCXCIII.

AHI defines financial forecasting and multivariate forecasting analyses as feed-forward topology that converges to better local optima.

Desideratum CCCXCIV.

AHI aggregates financial and economic data across multiple instruments to enable richer sets of data; allowing AHI to simultaneously train single market models from many signals.

Desideratum CCCXCV.

AHI uses time series lags, moving averages, and moving correlations, to generate market memory and market movement.

Desideratum CCCXCVI.

AHI uses back-propagation learning algorithms and mini-batching to solve computationally intensive equations in matrix form. Once expressed in matrix form, optimized linear algebra routines map out hedging algorithms.

Desideratum CCCXCVII.

AHI defines the differences between statistical manifolds, three dimensional manifolds, amplitudes, and knots, as the parameters for geometrically mapping the probability of trades.

Desideratum CCCXCVIII.

AHI assumes space and time have their own axes for probabilistic analysis in vertical time.

Desideratum CCCXCIX.

AHI defines trades as dependent space-time knots.

Desideratum CD.

AHI defines the *truth* as binary and compositional. *Uncertainty* is defined as ternary and compositional.

Desideratum CDI.

AHI defines weights attached to trades with two meanings: either they are truth values or they are degrees of confidence. Thus, logical trades are fuzzy; since, *truth* is a matter of degree. The algebraic settings for fuzzy trades are not Boolean. *Truth* is not binary and weights express the inability for hedging algorithms to learn whether a trade is true or false. Uncertainty is due to incomplete data. *Grading* the *truth*, refers to many valued variabilities.

Desideratum CDII.

AHI assumes the compositional logic of uncertainty leads to the mathematical difficulty of maintaining a Boolean setting. The uncertainty of the formula is not a function of the uncertainties of its constituents.

Desideratum CDIII.

AHI assumes probability functions are not an efficient representation of incomplete data.

Desideratum CDIV.

AHI defines degrees of probability as extensions of binary truth values; provided that the probabilistic data for the algorithms are correct.

Desideratum CDV.

AHI assumes that the degrees of probability lie on meta-level Boolean trades that define the paradigm of partial truths and grades of uncertainty, as a single framework.

Desideratum CDVI.

AHI defines *trading* as: set E represents a fuzzy set of possibility distribution π over frames of discernment. The set of E possible states, is ordered in terms of normality and plausibility. $\pi(\omega)$ reflects the extent of the state of affairs. Range V of π represents the unit interval. The consistency of data is ensured if $\pi(\omega) = 1$ for some ω.

Desideratum CDVII.

AHI defines the distribution of π on Ω and the set of [p], as fuzzy trades. Proposition p for truth scale L uses membership grades $\mu[p]\omega$. The plausibility scale that uses possibility degrees $\pi(\omega)$ for state of affairs ω, is the hedging strategy.

Desideratum CDVIII.

AHI assumes that probabilities do not define logical entailment. *If* a set of trades are true, their probability is greater or equal to a specific threshold; which is not deductively closed. Probabilistic reasoning does not maintain probability bounds across inference steps.

Desideratum CDIX.

AHI uses the probabilistic counterpart of resolution rules when local computation tools for computing the lower bounds of probability, form probabilistic constraints.

Desideratum CDX.

AHI assumes that conditional constraints are not defined by the weights attached to classical formulas; since the conditioning bar is not a Boolean logical connective.

Desideratum CDXI.

AHI assumes that due to syntax-equivalence, possibility and necessity measures are defined as sets.

Desideratum CDXII.

AHI defines *belief revision* as conditional possibility measures in qualitative settings that obey Bayesian properties.

Desideratum CDXIII.

AHI assumes possibilistic *conditioning* differs from the possibility of a trade implication.

Desideratum CDXIV.

AHI defines first order possibilistic logic as formulas and weights of certainty and priority.

Desideratum CDXV.

AHI defines weighted logical formulas with conjunctive forms of equivalent sets of clauses.

Desideratum CDXVI.

AHI defines possibility distributions that rank-order market phases (possible worlds) according to their level of plausibility or priority. Thus, there are direct translations of one representation to the other.

Desideratum CDXVII.

AHI defines *triangular norms* as associative, non-decreasing symmetric operations.

Desideratum CDXVIII.

AHI uses possibilistic logic as a deductive system.

Desideratum CDXIX.

Possibilistic logic where weights are attached to formulas, are replaced by fuzzy sets that support the truth value of the formula. *Timed* possibilistic logic, where weights are attached to formulas, are replaced by fuzzy sets of *time* when the formula is true. *Logic* supporters with weights attached to formulas are replaced by sets of *irredundant subsets* of assumptions that support the formula.

Desideratum CDXX.

AHI defines probability with the following parameters:

Mass: (10^{24} kg) 5.9723

Volume: (10^{10} km^3) 108.321

Equatorial Radius: (km) 6378.137

Polar Radius: (km) 6356.752

Volumetric Mean Radius: (km) 6371.000

Core Radius: (km) 3485

Ellipticity: 0.003353

Mean Density: (kg/m^3) 5514

Surface Gravity: (m/s^2) 9.798

Surface Acceleration: (m/s^2) 9.780

Moment of Inertia: (I/MR^2) 0.3308

$J_2(x\ 10^{-6})$: 1082.63

Surface Pressure: (mb) 1014

Surface Density: (kg/m^3) 1.217

Scale Height: (km) 8.5

Average temperature: 288 K, 15 C

Diurnal Temperature range: 283 K– 293 K, 10 C- 20 C

Mean molecular weight: 28.97