

Homeostat's Units and CTRNN's Neurons

Annotated Edition with Corrections

The following reproduces the original text of the research note, with inline corrections appended after each passage that requires amendment. Two summary sections appear at the end.

1 State equations of a CTRNN unit

Canonical state equations of a CTRNN's neuron in a network of N elements are [BEER & GALLAGHER 1992], [DI PAOLO 2000]:

$$\tau_i \dot{y}_i = -y_i + \sum_{j=1}^N w_{ji} z_j + I_i \quad (1)$$

$$z_j = \frac{1}{1 + e^{b_j - y_j}} \quad (2)$$

where τ is the time constant ($= RC$ in the equivalent circuit) that allows for temporal summation of the incoming input, y_i represents the cell potential of neuron y , or the *value* of the neuron, w_{ji} is the weight of the connection from neuron j to neuron i , and I_i is the incoming input, which is set to 0 for all non-input neurons. The z function is the ordinary sigmoid function that converts the potential of the neuron into a value in the interval $[0, 1]$ as a function of the neuron's potential, y , and the neuron's bias, b . The bias term is the parameter in the neuron's state equation that allows for spatial summation of inputs.

Correction. (a) The sigmoid in eq. (2) can be rewritten as $z_j = \frac{1}{1 + e^{-(y_j - b_j)}}$. Beer's standard convention uses $\frac{1}{1 + e^{-(y_j + \theta_j)}}$, so the document's $b_j = -\theta_j$. The parameter b as defined here is a *threshold* (higher b = harder to activate), not a bias in Beer's sense (higher θ = easier to activate). Calling it "bias" may confuse readers who consult Beer's papers directly.

(b) The claim that "the bias term is the parameter in the neuron's *state equation* that allows for spatial summation of inputs" is doubly incorrect. First, b does not appear in the state equation (1) at all—it appears only in the output equation (2). Second, b does not perform spatial summation; the $\sum_j w_{ji} z_j$ term does that. The bias (or threshold) shifts the activation curve, determining at what level of net input the neuron begins to fire appreciably.

[Footnote in original:] This equation was originally proposed by Hopfield [HOPFIELD 1984], on the basis of an analogy between neural networks and RC electronic circuits, and expressed in terms of resistance, capacitance, and voltage potential, input current, (R, C, V, I) for a neuron i as: $C_i \dot{u}_i = -\frac{u_i}{R_i} + \sum_j T_{ij} V_j + I_i$, where the summative term represents the weighted inputs to neuron i , the second term represents transmembrane resistance, and the final term, I_i , is the externally applied input to the neuron. Normalizing with $\tau_i = R_i C_i$ leads to eq. (1).

Correction. The normalization step is incomplete as stated. Dividing Hopfield’s equation by C_i and substituting $\tau_i = R_i C_i$ yields:

$$\tau_i \dot{u}_i = -u_i + R_i \sum_j T_{ij} V_j + R_i I_i$$

Reaching eq. (1) requires *also* absorbing R_i into the weights ($w_{ji} = R_i T_{ji}$) and into the input ($I_i^{\text{new}} = R_i I_i^{\text{old}}$). This is standard practice but should be stated.

2 State equation of a Homeostat unit

Ashby describes the behavior of the 4-unit Homeostat as follows in *Design of a Brain* [ASHBY 1960, p. 247]:

$$\frac{dx_i}{dt} = \dot{x}_i$$

$$\frac{d}{dt}(m\dot{x}) = -k\dot{x}_i + l(p - q)(a_{i1}x_1 + \dots + a_{i4}x_4) \quad (3)$$

Correction. The left-hand side of eq. (3) is missing the subscript i . It should read $\frac{d}{dt}(m\dot{x}_i)$, i.e. $m\ddot{x}_i$ for constant m .

where x_i is the i -th unit’s needle’s angle of deviation, m is its moment of inertia, p and q are the potentials at the opposite ends of the fluid-filled trough, k represents the frictional forces acting on the needle’s movement, and l “depends on the valve,” i.e. it is a gain parameter. Rewriting (3) in notation closer to CTRNN’s would give us, for a Homeostat of N units:

$$m\ddot{x}_i = -k\dot{x}_i + l(p - q) \sum_{j=1}^N a_{ij}x_j \quad (4)$$

Correction. Note the weight subscript convention: in eq. (1), w_{ji} denotes the weight *from* j *to* i (first subscript = source). Here in eq. (4), a_{ij} multiplies x_j , so it is the weight *from* j *to* i (first subscript = destination). Thus $a_{ij} \equiv w_{ji}$ —the subscript orderings are *opposite*. This is never mentioned in the text and can cause confusion when comparing the two systems term by term.

In Ashby’s device, the gain parameter, l , and the difference in potential, $p - q$, are the same for all units, thus allowing to slightly simplify the equation by assuming a constant gain factor $g = l(p - q)$:

$$m\ddot{x}_i = -k\dot{x}_i + g \sum a_{ij}x_j \quad (5)$$

My implementation substantially follows equation (4), with the difference that it does not assume l , p , and q to be the same across all units.

2.1 Numerical implementation of Ashby’s equation

[This subsection is not part of the original note. It documents how the Homeo simulator codebase actually implements eq. (4), based on a code review of the Core/ classes.]

The claim in the author’s note above is confirmed by the code. Furthermore, the Aristotelian / Newtonian distinction discussed in Section 3 is not merely theoretical—it is architecturally encoded in the class hierarchy. Three unit classes implement three different physics models:

Class	Physics	Equation
HomeoUnit (base)	Aristotelian	$v = F/m$; viscosity as a multiplicative attenuator of force
HomeoUnitAristotelian	Aristotelian	$v = (F + \text{drag})/m$; viscosity as drag proportional to force
HomeoUnitNewtonian	Newtonian	$ma = F_{\text{input}} - v \dot{x}$; Stokes-law drag proportional to velocity

The `HomeoUnitNewtonian` class—the one used in all GA experiments and robotic simulations—implements eq. (4) through a four-step discrete-time integration with $\Delta t = 1$:

1. **Torque** (`HomeoUnit.computeTorque`). Computes the weighted sum of all incoming outputs:

$$\text{inputTorque} = \sum_j (\text{output}_j \times \text{switch}_{ij} \times \text{weight}_{ij} + \text{noise}_{ij})$$

This is the $\sum a_{ij} x_j$ term, with weights, polarity (switch), and connection noise.

2. **Drag** (`HomeoUnitNewtonian.stokesLawDrag`). Computes viscous friction as:

$$\text{drag} = -\text{viscosity} \times \text{currentVelocity}$$

This is the $-k \dot{x}_i$ term.

3. **Position update** (`HomeoUnitNewtonian.newLinearNeedlePosition`). Combines torque and drag into acceleration, then applies the standard kinematic equation:

$$\begin{aligned} \text{totalForce} &= \text{inputTorque} + \text{drag} \\ a &= \text{totalForce} / \text{mass} \\ x_{t+1} &= x_t + v_t + \frac{1}{2} a \end{aligned}$$

This is $x_{t+1} = x_t + v_t \Delta t + \frac{1}{2} a \Delta t^2$ with $\Delta t = 1$.

4. **Velocity update** (`HomeoUnitNewtonian.selfUpdate`). The new velocity is recovered from the displacement:

$$v_{t+1} = 2(x_{t+1} - x_t) - v_t$$

Since $x_{t+1} - x_t = v_t + \frac{1}{2} a$, this simplifies to $v_{t+1} = v_t + a$, i.e. $v_{t+1} = v_t + a \Delta t$.

Steps 3 and 4 together constitute the **velocity Verlet** integration scheme (second-order accurate in position), applied to the ODE

$$m \ddot{x}_i = -\text{viscosity} \cdot \dot{x}_i + \sum_j a_{ij} x_j$$

which is exactly eq. (4) with per-unit mass and viscosity, and with Ashby’s $l(p-q)$ absorbed into the per-connection weights.

Each unit holds its own instance variables for mass (`needleUnit.mass`), viscosity (`self.viscosity`), and self-connection weight (`self.potentiometer`), confirming that the implementation follows the general equation (4) with independent per-unit parameters, not the constant-gain simplification (5).

Integration step size and alignment with CTRNN practice. In the standard CTRNN literature, the forward Euler update

$$y_i(t+h) = y_i(t) + h \frac{1}{\tau_i} \left(\sum_j w_{ji} \sigma(y_j + \theta_j) + I_i - y_i(t) \right)$$

depends on an explicit step size parameter h , which controls the granularity of the discrete-time integration. A necessary stability constraint is $h < \tau_{\min}$: if $h \geq \tau_i$ for any unit, the update factor $(1 - h/\tau_i)$ goes negative and the integration diverges. Typical values in the literature are $h = 0.01$ – 0.1 with $\tau \geq 1$. Beer’s canonical C++ CTRNN library provides both a forward Euler step (`EulerStep`) and a 4th-order Runge–Kutta step (`RK4Step`), though forward Euler is used overwhelmingly in practice. DiPaolo uses forward Euler with 5 sub-steps per sensory-motor cycle to improve accuracy without changing method.

In the Homeo codebase, the equivalent parameter Δt is implicitly set to 1: there is no explicit step size variable, and each call to `step()` advances time by one unit. This is viable because (a) the velocity Verlet scheme is second-order accurate (vs. Euler’s first order), hence more tolerant of large steps; (b) the system is heavily damped (viscosity), suppressing instability; and (c) Ashby’s original parameters were tuned for unit timesteps on his physical machine.

An explicit Δt parameter could, however, be introduced into the Verlet integration to bring it in line with CTRNN practice. The modified update would read:

$$\begin{aligned} x_{t+\Delta t} &= x_t + v_t \Delta t + \frac{1}{2} a_t \Delta t^2 \\ v_{t+\Delta t} &= v_t + \frac{1}{2} (a_t + a_{t+\Delta t}) \Delta t \end{aligned}$$

Setting $\Delta t < 1$ and running multiple sub-steps per logical timestep would increase integration accuracy at the cost of proportionally more computation—the same trade-off that CTRNN practitioners navigate when choosing h . This would be particularly relevant if future experiments use lower mass values or stronger connection weights, both of which increase the effective stiffness of the system and thus demand finer temporal resolution.

2.2 Harmonic oscillators

For a slightly different take on the issue: notice that the homeostat’s units could also be seen as oscillators. Start with a single unit. If we provide a negative self-connection, and assume no frictional forces (i.e. the viscosity of the fluid in the trough is 0), then $-k\dot{x}$ in (5) becomes 0 and $g \sum a_{ij} x_i$ becomes simply $-ga_{jj}x$, with a_{ii} being the weight of the self-connection. Putting $ga_{ii} = k$, the unit’s behavior is described by

$$m\ddot{x} = -kx$$

which is the canonical equation for a harmonic oscillator.

Correction. (a) The expression “ $g \sum a_{ij} x_i$ becomes simply $-ga_{jj}x$ ” contains two subscript errors. For a single unit i with only a self-connection, the sum $g \sum_j a_{ij} x_j$ reduces to $g \cdot a_{ii} \cdot x_i$. The subscript should be a_{ii} (not a_{jj}), and the summation variable should be x_j (not x_i). Since the self-connection is negative ($a_{ii} < 0$), this equals $-g|a_{ii}|x_i$.

(b) The symbol k is being silently repurposed. In Ashby’s equation (5), k is the *friction coefficient* (multiplying \dot{x} , opposing velocity). Here, “putting $ga_{ii} = k$,” it becomes the *spring constant* (multiplying x , providing a restoring force). Then in the next paragraph, viscosity is introduced under a new symbol v . The reuse of k for a physically different quantity without comment is confusing. A distinct symbol (e.g. $\kappa = g|a_{ii}|$) would be clearer.

If we add a viscosity factor v due to the trough’s fluid and assume it proportional to velocity, we obtain the canonical equation for a damped oscillator: $m\ddot{x} = -v\dot{x} - kx$. Thus, the basic building block of

a Homeostat is a damped harmonic oscillator. And a Homeostat (leaving uniselectors' action aside) is simply a collection of variously coupled harmonic oscillators.

On the other hand, in a “real-world” translation of the homeostat we can assume that the real-world inputs the units are connected to are not self-regulating, i.e. they are not (damped) harmonic oscillators. It follows that a *single* negatively self-connected unit connected to the “real” environment is equivalent to a driven harmonic oscillator:

$$m\ddot{x} = -v\dot{x} - kx + aF(t)$$

where $F(t)$ and a are, respectively, the input the unit is receiving from the environment and the weight it assigns to that input. A “real world” Homeostat (or its equivalent homeostat simulation, if possible) is thus a collection of driven harmonic oscillators (uniselectors' actions aside).

3 Comparison between the two

The comparison between (1) and (4) can be carried out at three levels: the general shape of the description (the model), the terms involved, and the relation between a unit's value (i.e. potential and rotation angle, respectively, for CTRNNs and Ashby) and a unit's output.

3.1 The mathematical model

Here is where we find the greatest difference between the two. Although the models are remarkably similar, the important difference is that the value y (potential) of a CTRNN unit is defined in terms of its *first* derivative, whereas the value x of an Ashbian unit is expressed in terms of its *second* and first derivatives.

Mechanically interpreted in traditional notation: $F = ma$ or $m\dot{v} = F$, and $v = \dot{s}$, with the traditional abbreviations for mass, acceleration, velocity and displacement, where the force F acting on the object can be further decomposed into a reactive frictional force proportional to velocity plus the active forces acting on the object (assuming the simplest mathematical model of friction). Thus the unit is fundamentally modeled as a Newtonian object rotating on a friction-affected pivot, and the basic parameters affecting its value (angular position) are its mass (or moment of inertia) and the friction it exerts (assuming the difference in potential is a constant; otherwise it would have one additional parameter): $m, k, (p-q)$. If we try to interpret the CTRNN unit mechanically, for the sake of comparing it to the Homeostat, we see that (1) can be read as the model of an *Aristotelian* object, that is, an object whose velocity is proportional to the forces it is subjected to. In other words, the CTRNN unit is normally at rest in absence of incoming forces (its biological equivalent, the real neuron, having an appropriately called resting potential), whereas a Homeostat's unit in the same situation is in uniform motion.

The problem is that a straightforward “Aristotelian” dynamical model would have velocity equal to force over mass ($F = mv$ or, rather, $v = \frac{F}{m}$; in differential terms: $\dot{m}s = F$).

Correction. “ $\dot{m}s = F$ ” should be “ $m\dot{s} = F$.” The time derivative applies to the displacement s , not to the mass. The intended expression is $m\dot{s} = F$, i.e. force equals mass times velocity—the Aristotelian model.

Now, assuming, in the CTRNN case, that the forces are equal to the incoming inputs plus the externally applied current, and using y to represent the mechanical displacement, an Aristotelian model would give us the following equation: $m\dot{y} = \sum_j w_{ij}y_j + I_i$. Comparing this equation with eq. (1) makes the problem clear: we are missing a term. Namely, $-y_i$. If we rewrite the equation according to its original formulation, namely as $C_i\dot{y}_i = -\frac{y_i}{R_i} + \sum_j w_{ij}y_j + I_i$, we can perhaps make some progress. Now this shows, interpreted mechanically, that the applied force has three components, one of which is proportional to displacement y and to a resistive factor R , and the other two are dependent on the directly applied forces. This is not quite the original Aristotelian dynamical model, which explains friction as proportional to mass (and therefore contained in the C_i term in the equation). Furthermore, having a resistive force proportional

to displacement models, in mechanical terms, the resistance encountered when driving a stake through the ground, i.e. the resistance of an extended object entering a viscous fluid. I suppose what I am saying is that the electro-mechanical device that would (approximately) realize the CTRNN equation and be as close as possible to Ashby's contraption would be made up of

1. highly viscous “wells” (with an electric potential between surface and bottom) with “stakes” being driven in and out of them. This part would implement the $-\frac{y_i}{R_i}$ term.
2. very heavy friction in the mechanism's joints, surfaces, etc. This would approximate the Aristotelian modeling of force being proportional to velocity and not to acceleration, and thus the $C_i \dot{y}_i$ component instead of the $m \ddot{y}_i$ component in Ashby's model.

Now the second big difference is that there is a linear function (in fact, the identity function) in Ashby's model, whereas there is a non-linear function, the logistic one, in the CTRNN case.

The main difference, of course, is that whereas Ashby's Units have an identity function from unit's value to output, CTRNN units have a sigmoid, hence a non-linear, though continuously differentiable function. The reason behind the adoption of a sigmoid nonlinear function is due to the wish to model the frequency-coding properties of real neurons, since such a function converts, over a short time span, a neuron's value into a firing frequency.

3.2 Term to term

Now for the equivalence with Ashby's Homeostat. Its units are electro-mechanical contraptions where the values of interest to the modeler are the physical properties of the constantly moving parts and the forces acting on them. The state of a unit is thus represented by the velocity of the needle, the mechanical force acting on it (that is, the torque produced by the magnets coiled around its pivot as a result of the incoming currents from the other units), the mass of the needle, and the resistance the needle encounters in its movement through the fluid-filled trough. A term-by-term equivalence can perhaps be found by interpreting the capacitance C_i and resistance R_i of the CTRNN model as being, roughly, functionally equivalent to, respectively, mass and friction in Ashby's model.

In mechanical terms, since the decay constant τ represents the neuron's resistance to a change of state, it is equivalent to the needle's mass providing an inertial resistance against change of velocity. The bias term b represents a neuron's resistance against firing and is equivalent to the frictional resistive forces acting against the needle's displacement. The weights of the incoming connections, w_{ji} , are obviously equivalent to the various resistors' values on the incoming currents, while the incoming current, I_i , is normally equal to 0 unless the Homeostat's operator decides to manually manipulate one or more of the units (as in Ashby's various experiments), in which case it corresponds to the external force acting on the needle.

Correction. The mapping of the bias b to friction is incorrect. In the CTRNN, the bias b (or threshold, in this document's sign convention) shifts the sigmoid activation curve—it determines at what level of net input the neuron begins to fire. It has no direct mechanical analog in Ashby's device.

The term that is functionally analogous to friction/damping is the $-y_i$ leak term in eq. (1) (or $-y_i/R_i$ in Hopfield's original form): it pulls the neuron's state back toward rest, just as $-k\dot{x}_i$ pulls the needle's velocity toward zero. Of course the analogy is imperfect—the leak opposes *displacement* while friction opposes *velocity*—which is precisely the first-order vs. second-order (Aristotelian vs. Newtonian) distinction correctly identified earlier.

A more defensible term-by-term mapping would be:

CTRNN	Homeostat
τ_i (time constant = $R_i C_i$)	m_i (mass / moment of inertia)
$-y_i$ (leak / passive decay)	$-k\dot{x}_i$ (viscous friction)
w_{ji} (connection weights)	a_{ij} (connection weights)
I_i (external input current)	external force on needle
b_j (sigmoid threshold)	<i>no direct analog</i>

4 Conclusion

Since in the original Homeostat all units are constructionally identical, the only free parameters governing the evolution of the network are the weights w_{ij} controlling the connections between the units. If, on the other hand, we consider the possibility of a “generalized” Homeostat (rather easily realizable through a computer simulation, continuous or discrete), we should consider all the parameters in eq. (4) as being independent, as we can easily construct units with different masses, different viscosities, potentials, gains, etc. That means that the simulation (and training, perhaps via GA techniques) of a homeostat of j units would entail the determination of 4 parameters per unit (mass, viscosity, gain, and potential) + j weights, totaling $j(j+4)$ network parameters. This number is actually not very different from the CTRNN standard of $j(j+2)$ (weights, plus time constant and bias).

What is wrong

The following substantive errors affect the mathematical or conceptual content of the note.

1. **Bias b mislocated and misdescribed.** The text states that the bias is “the parameter in the neuron’s state equation that allows for spatial summation of inputs.” In fact b appears only in the output (sigmoid) equation (2), not in the state equation (1). It does not perform spatial summation (which is done by $\sum w_{ji}z_j$); it shifts the activation threshold.
2. **Sigmoid sign convention opposite to Beer’s.** The document defines $z_j = \frac{1}{1+e^{b_j-y_j}}$, making b_j a *threshold* (higher b = harder to activate). Beer’s standard parameterization uses $\frac{1}{1+e^{-(y_j+\theta_j)}}$, making θ_j a *bias* (higher θ = easier to activate). Since $b_j = -\theta_j$, calling b a “bias” without comment is misleading.
3. **Missing subscript in Ashby’s equation.** The left-hand side of eq. (3) reads $\frac{d}{dt}(m\dot{x})$ but should be $\frac{d}{dt}(m\dot{x}_i)$, since the right-hand side is indexed by i .
4. **Wrong subscripts in the harmonic oscillator derivation.** The text writes “ $g \sum a_{ij}x_i$ becomes simply $-ga_{jj}x$.” Both subscripts are wrong: for a self-connection of unit i , the sum $g \sum_j a_{ij}x_j$ reduces to $g \cdot a_{ii} \cdot x_i$. The correct statement uses a_{ii} (not a_{jj}) and x_j in the sum (not x_i).
5. **Symbol k silently repurposed.** In Ashby’s equation, k is the friction coefficient (multiplying \dot{x}). In the oscillator derivation, k becomes the spring constant (ga_{ii} , multiplying x). Viscosity is then introduced under a new symbol v . The reuse of k for a physically different quantity is never flagged.
6. **Derivative on wrong symbol.** The “Aristotelian” model is written as “ $\dot{m}s = F$ ” but should be “ $m\dot{s} = F$ ” (the time derivative applies to the displacement, not the mass).
7. **Bias mapped to friction in the term-by-term comparison.** The text claims the bias b “is equivalent to the frictional resistive forces acting against the needle’s displacement.” This mapping is incorrect. The $-y_i$ leak term (or $-y_i/R_i$ in Hopfield’s form) is the functional analog of friction/damping. The bias b shifts the sigmoid threshold and has no direct mechanical counterpart in Ashby’s device.
8. **Hopfield-to-Beer normalization elides a step.** The footnote states that “normalizing with $\tau_i = R_i C_i$ leads to eq. (1).” In fact, reaching eq. (1) also requires absorbing R_i into the weights ($w_{ji} = R_i T_{ji}$) and into the input.
9. **Weight subscript conventions are opposite and uncommented.** The CTRNN equation uses w_{ji} (first subscript = source), while the Homeostat rewrite uses a_{ij} (first subscript = destination). Both denote the weight from j to i , but the index ordering is reversed. A comparative note never mentions this.

What is correct

1. **The Aristotelian vs. Newtonian distinction.** The central insight of the note—that the CTRNN is a first-order (velocity \propto force, “Aristotelian”) system while the Homeostat is a second-order (acceleration \propto force, Newtonian) system—is correct and illuminating. It concisely captures the deepest structural difference between the two architectures.
2. **The harmonic oscillator interpretation.** A single Homeostat unit with a negative self-connection and no friction is indeed a harmonic oscillator; with viscosity, a damped harmonic oscillator; with external input, a driven damped harmonic oscillator. The conclusion that the Homeostat is a system of coupled oscillators follows correctly.
3. **CTRNN and Hopfield equations.** The CTRNN state equation (1) and the Hopfield equation in the footnote are correctly stated and correctly related (modulo the elided normalization step).

4. **Ashby's equation and its rewrite.** Equations (4) and (5) are correct renderings of Ashby's dynamics in modern summation notation.
5. **The mapping $\tau_i \leftrightarrow m_i$.** Identifying the CTRNN time constant (capacitance \times resistance) with the Homeostat's mass (inertia resisting change) is the standard and correct electrical–mechanical analogy.
6. **The parameter count.** The calculation that a generalized Homeostat of j units has $j(j + 4)$ free parameters, versus $j(j + 2)$ for a CTRNN, is correct: 4 unit-level parameters (mass, viscosity, gain, potential) plus j^2 weights, versus 2 unit-level parameters (time constant, bias) plus j^2 weights.
7. **Identity vs. sigmoid output function.** The observation that Ashby's units use an identity output function while CTRNNs use a sigmoid, and that this sigmoid models frequency coding in real neurons, is correct.
8. **The “driven oscillator” real-world interpretation.** The argument that a Homeostat unit connected to non-self-regulating real-world inputs behaves as a driven harmonic oscillator is physically sound.