Automated Calculation Methods Using Classical Machine Design Theory

Tristan Lake Harris Caja

Arkansas Tech University

2024

Table of Contents

Three-phase Induction Motors

Perhaps the most common type of electric motor is the polyphase induction motor, rich in both history and literature. These motors are self-starting and reliable, yet limited only to where polyphase power is available. Polyphase motors have a uniform revolving field, which is produced by the stator alone, meaning that all voltages induced in the secondary are due to speed action and none to transformer action. Three-phase induction motors are the most common polyphase motor, and therefore will be the choice of motor discussed herein.

Standard Connections

Two of the most common types of three-phase motor connections are the wye and delta connections. Each connection provides different characteristics for the voltage and current in each phase, and standardized terminal markings for these connections will be discussed in this section.

Wye-connected Three-phase Motors

In the conduit boxes of wye connected motors, all three phases meet at the "star" or neutral point. In this connection, the voltages in each phase are larger than the line voltages by a factor of $\sqrt{3}$. The currents in each phase however are the same as that of the line. Occasionally motors will have neutral lead T0 brought out to account for imbalances. When desired, each phase can be separated into its own section by using 6 leads. Terminal markings of this type are shown below.

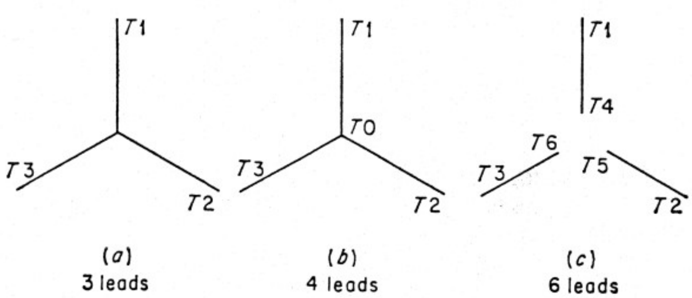

Fig.	L1	L2	L3	Open	Join
(a)	T1	T2	T3	—	—
(b)	T1	T2	T3	T0	—
(c)	T1	T2	T3	—	T4,T5,T6

Figure 1, single-voltage, wye-connected terminal markings, given by Veinott[3].

Delta-connected Three-phase Motors

Delta-connections of three-phase motors differ from wye-connections in that the voltages in each phase are the same as that of the line, while the phase currents are *reduced* to $1/\sqrt{3}$ of line current. Conversely, this means that phase current is increased by a factor of $\sqrt{3}$, drawing more current from the line and thus giving higher torques.

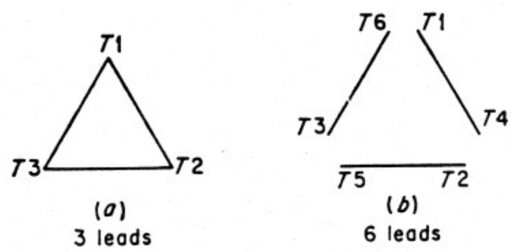

Fig.	L1	L2	L3
(a)	T1	T2	T3
(b)	T1, T6	T2, T4	T3, T5

Figure 2, single-voltage, delta-connected terminal markings; Veinott[3].

Terminal markings for delta-connected motors are shown above. If it is desired, each phase can be designated as its own section by using 6 leads.

Dual-voltage Motors

When motors are rated to operate on two different voltages, the windings for each phase must be split into two sections by some means to ensure that approximately the same current flows on either connection. There are several ways to split each phase, but perhaps the most common way of doing so in three-phase motors is to have each section contain half the number of poles. Terminal markings for a dual-voltage wye-connected motor are shown below, where each section represents half the poles for its respective phase.

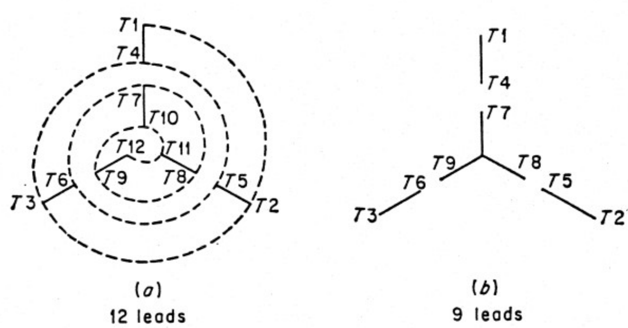

Fig.	Voltage	L1	L2	L3	Join			
(a)	High	T1	T2	T3	T4, T7	T5, T8	T6, T9	T10, T11, T12
	Low	T1, T7	T2, T8	T3, T9	T4, T5, T6		T10, T11, T12	
(b)	High	T1	T2	T3	T4, T7	T5, T8	T6, T9	———
	Low	T1, T7	T2, T8	T3, T9	T4, T5, T6			

Figure 3, dual-voltage, wye-connected terminal markings, given by Veinott[3].

Irregular Windings and Special Connections

While not given extensive coverage here, it is important to consider the properties of certain special connections. Most notably, the consequent-pole connection is of interest. In all the connections discussed heretofore, currents in each pole are wound for opposite directions, giving different polarities when excited. In the consequent-pole connection, all poles are wound in the same direction to give the same polarities. As a *consequence*, magnetic poles of the opposite polarity develop between the pole groups. Due to these consequential poles, these motors will generally have twice as many poles as there are pole groups.

Summary

A brief overview of typical three-phase motor connections was the primary focus of this section. As this type of motor has a self-starting revolving field that induces current in the secondly solely from cutting action, a qualitative explanation was superfluous. A quantitative analysis of the three-phase motor will be the primary focus of the following section.

Quantitative Analysis of Three-phase Induction Motors

Three-phase Motor Action

An important difference between the induction-run single-phase motor and the polyphase motor is that rotor currents are induced solely from speed action; the cutting of flux by the rotor bars. The rotor begins to move in the same direction as the field, owing to the interactions between the primary revolving field and the field from the secondary currents. It becomes apparent that as the rotor comes up to speed, the relative motion between the revolving field and the rotor decreases, and that less primary flux is cut by the rotor bars.

The relativity between the synchronous speed of the revolving field and the speed of the rotor is called the slip. It is a measure of how fast the lines of flux cut the rotor bars, and therefore the relative motion between the two:

$$s = 1 - \left(\frac{rotor\ speed}{sync.speed}\right) \qquad (1)$$

Note this value ranges from 1 to 0 as the motor increases in speed, meaning that at standstill, s = 1.0, and at no-load, s ≈ 0.0.

One may be curious as to how a revolving field is produced by the stator. Afterall, the primary winding is made up of stationary coils. However, the currents in each phase are displaced by 120 degrees in time. Since flux lines can never cross, the resulting field components will only combine, causing the magnetic field to rotate from the pulsations in current.

The Equivalent Circuit

In balanced polyphase motors, the input supply of voltage and current will be the same for every phase. Hence the motor can be represented by a single phase, as shown below.

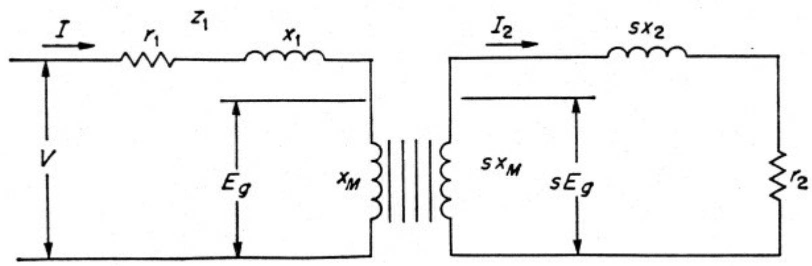

Figure 4, the polyphase equivalent circuit; Veinott[3].

The coupled circuits in the figure above are both closed, with the left representing the primary, the right the secondary. The primary current I sets up a magnetic field due to both the leakage flux, depicted as x_1, and the mutual flux x_M, which links the primary and secondary windings. The air-gap voltage E_g is the voltage drop across the latter reactance and is an indicator of the strength of the revolving field. It can be shown here that the primary impedances drops, given by z_1, will reduce the field strength, or voltage drop across E_g.

At locked-rotor*, the voltage induced in the secondary is E_g. But as the motor comes up to speed, the relative motion, and thus the slip, decreases, causing a reduction in induced voltage. Likewise, the frequency of the secondary current is *sf*, and the secondary reactance must then be decreased by the slip as well, as inductive reactance is proportional to frequency. These concepts are shown by sE_g, sx_M, and sx_2 in the circuit.

The equation for the secondary circuit can be arranged to give

$$I_2 = \frac{sE_g}{r_2 + jsx_2} \qquad (2)$$

where

$$E_g = j(I - I_2)x_M \qquad (3)$$

The circuit may be greatly simplified while keeping its validity by dividing the numerator and denominator of Eq. (2) by 1/s, making the secondary circuit equation

* Or at standstill.

$$I_2 = \frac{E_g}{\left(\frac{r_2}{s}\right) + jx_2} \qquad (4)$$

Another equivalent circuit, which uses Eq. (4) for the secondary, is given below, where additional resistor r_{fe} has been paralleled with the mutual reactance to represent core loss. This network is a historical, commonly used model for predicting the performance of polyphase induction motors.

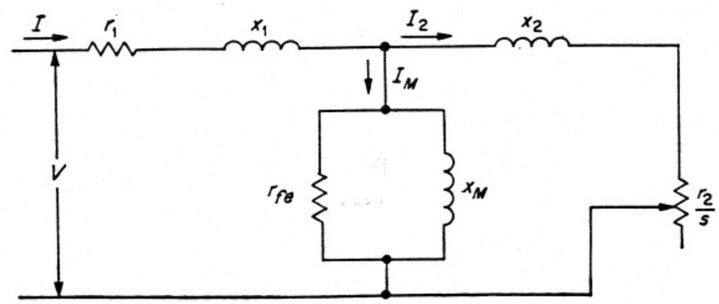

Figure 5, turns-transformation of the equivalent circuit[3].

Determination of Torque

It is intuitive that the power transferred across the air gap is the secondary input power. Neglecting secondary leakage, it can be shown that

$$Sec.Input = \frac{mI_2^2 r_2}{s} \quad [watts] \qquad (5)$$

where m is the number of phases. The secondary output, a mechanical power developed, is the secondary input less the I^2r losses:

$$Sec.Output = m\left(I_2^2\frac{r_2}{s} - I_2^2 r_2\right) = \frac{mI_2^2 r_2}{s}(1-s) \quad [sync.watts] \qquad (6)$$

This power output may then be converted into the torque developed:

$$Torque = \frac{7.04}{rotor\ speed} \times \frac{mI_2^2 r_2}{s}(1-s) \quad [lb-ft] \qquad (7)$$

Or equivalently

$$Torque = \frac{7.04}{sync.speed} \times \frac{mI_2^2 r_2}{s} \quad [lb-ft] \qquad (8)$$

This shows that the resistive loss in the secondary branch of the circuit is equivalent to the torque developed by the motor. This torque must supply friction and windage losses, stray-load losses, and the load.

Along with the slip s, the other variable in the equation for torque is the secondary current, which varies with slip and affects the total current drawn by the motor circuit.

Motor Currents

It will be shown in this section that currents in both the primary and secondary change at different load points, affecting the mutual flux, leakage flux, and hence the torque developed at varying points of operation.

Locked-rotor Currents

At the locked-rotor condition, $s = 1$, and the apparent resistance of the rotor is the same as the actual resistance. This implies that secondary resistance is at a minimum, and high currents flow in the rotor. Hence high currents flow in the stator too, and the primary impedance drop is high, decreasing the voltage drop across the air gap. Even so, there is an appreciable torque developed, owing to the strong secondary flux from high rotor currents interacting with the revolving field.

Full-load Currents

At full-load, the slip for common induction motors is approximately 0.05, meaning the apparent rotor resistance is some 20 times the actual resistance. Much less current is induced in the secondary than at locked-rotor, and this current flows at a high power factor, as the secondary circuit is mostly resistive.

It will help to remember that the primary current consists of a secondary component I_2 and a magnetizing component I_M, where I_2 is much smaller at full-load than at locked-rotor conditions due to the higher apparent resistance of the rotor.

No-load Currents

At no-load, $s \approx 0$, and the value of the variable resistor is infinite, indicating that the secondary circuit is open. Furthermore, since $r_{fe} >> x_M$, the core loss branch is practically short-circuiting the primary current through the mutual reactance. Therefore, the primary current all goes to supplying the revolving field. Power factor is low, as the motor circuit is predominantly reactive, with $x_M + x_1$ being much greater than r_1.

In actuality no-load speed could never be obtained by the rotor as friction, windage, and stray-load losses must be supplied (these act as a mechanical drag on the rotor). For that reason, there will always be a minute amount of secondary current and torque at sub synchronous speeds.

Summary

In dealing with motors, the generated torque is of primary interest. Torque in asynchronous motors is mainly of function of both speed and current. Values for the impedances of the equivalent circuit are but motor constants, and do not change like the slip and the current drawn.

Off this principle, one could see that conductor losses (I^2R) in the rotor and stator would follow the same general shape of the line current when plotted against speed. Core losses would generally increase with speed, as more of the current drawn flows across the core-loss and magnetizing branch.

Variables for both performance and losses will be expanded upon when evaluating an example: a polyphase equivalent circuit written in Python code. Analytical methods for predetermining motor constants are not complex, but still outside the scope of this paper. A numerical approach to doing so can be done using the finite element method[1, 2].

Revolving-field Theory

It was long ago when mathematicians discovered that a stationary pulsating wave could be resolved into two oppositely revolving waves, each moving at uniform speed and each having half the magnitude of the pulsating wave.

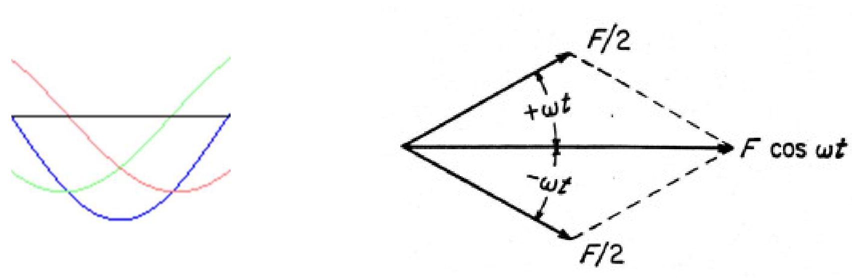

Figure 6, vector form of the revolving fields and their summation[3].

The algebraic sum of the revolving waves will give the pulsating wave, and conversely the pulsating wave can be decomposed into the two revolving waves.

The concept is more easily realized when the waves are represented as vectors. The figure to the right shows these waves in vector form. Two vectors, each with magnitude F/2, revolve in opposite directions at angular velocity ω. It can be shown that the sum of the revolving fields is given by Fcos(ωt).

At time t = 0, the vectors overlap and their addition would give twice the magnitude of either component vector. At time t = t, the revolving vectors would be displaced by ωt radians in their respective directions. When the vectors point in opposite directions, their sum is zero, and there is no completion of the parallelogram. Following this, the revolving vectors would give a pulsating field magnitude in the opposite direction, and this process continues.

Single-phase Motor Types

Single-phase induction motors are extremely common, being used in household appliances, power tools, fans, heaters, etc. While there are many variations of this kind of motor, which differ in starting methods, for purposes here they will be differentiated as follows:

- Induction-run: Single-phase motors that do not have the starting winding in the motor circuit at normal operating conditions, and
- Capacitor-run: Single-phase motors that keep the starting winding in the circuit at normal operating conditions.

Most of the discussion of the revolving-field theory herein will be concerned with the former, although the theory equally applies to capacitor-run motors.

Conditions for Motor Action

All induction motors utilize a revolving electromagnetic field set up by the stator to create torque. In considering a two-phase motor, there are two requirements necessary to produce a revolving field and thus motor action:

1. Two distinct phase windings which are 90 electrical degrees apart in space.
2. Two AC currents in each winding which are equal in magnitude but displaced 90 degrees in time.

Single-phase induction motors are therefore special cases of two-phase motors, where a voltage source is split across two windings. The main or “running” winding stays in the motor circuit throughout all operation, while the auxiliary or “starting” winding is temporarily paralleled with the main winding to produce the necessary starting torque.

Single-phase Motors without Starting Windings

A single-phase motor, without a starting winding, will set up a stationary pulsating field. At standstill, the flux of this pulsating field does in fact induce voltages in the rotor bars via transformer action, but the result is zero net torque, as secondary flux on opposite sides of the rotor tend to cancel out each other’s force.

It can be shown experimentally however that when external starting torque is supplied mechanically to the rotor, the motor will come up to speed on its own, even though there is no revolving field set up by the stator. At first, this may seem to violate the conditions for motor action, but this is the principle behind single-phase induction-run motors and the basis for the following discussion of revolving-field theory.

Application of the Revolving Field Theory to Single-phase Motors without Starting Windings

When applying the revolving-field theory to a stationary pulsating field in a single-phase motor, it helps to imagine two rotating magnetic fields in its place; each half the magnitude of the pulsating field and each moving in a different direction at synchronous speed. The forward field will move in the direction of the rotor and the backward field in the opposite direction. In physical reality, flux lines cannot cross, so these two rotating magnetic fields occupying the same space must be abstractly pictured as superimposing on one another.

Locked-rotor Torque

If the rotor is not moving, the forward and backward waves are equal, and the torques generated by each wave cutting the rotor bars are canceled by each other. Single-phase motors without a starting winding develop no locked-rotor torque, as confirmed by the theory.

No-load Conditions

At no-load, the rotor is moving almost as fast as the forward revolving wave, and little voltage is induced in the rotor, hence low secondary currents and I^2r losses. The backward field, however, cuts the rotor bars at nearly double the synchronous speed, since the slip between the rotor and the backward wave is nearly 2 (breaking action). This induces high voltages in the rotor bars, causing high secondary currents. But these currents induced by the backward field are in a direction as to suppress the backward field flux that induced them, damping out the backward field wave almost completely. The back-emf induced in the stator winding by the backward field is thus very small, and since the sum of the voltages induced by the two waves in the stator must equal the line voltage, the primary current increases until the forward wave becomes nearly equal to the magnitude of the pulsating wave. Therefore, at no-load the pulsating wave will be resolved into a strong forward wave and a very weak backward wave. The result is an elliptical field that varies in magnitude as it rotates. Due to the backward wave, single-phase motors then have considerable I^2r losses at no-load than polyphase motors, and a higher slip due to the pull of the backward torque.

Summary

The abstract nature of the revolving field theory allows for an effective method of predicting the performance of single-phase induction motors. There may be a forward and backward field from the main-winding alone, or resultant forward and backward fields from the addition of the forward and backward waves of the main and auxiliary windings, as each winding produces two fields. This concept will be developed further starting with the forward and backward fields when only the main winding is in the circuit.

Quantitative Analysis of Revolving-Field Theory

Main Winding-Only

Most single-phase induction-run motors will have a centrifugal switch that will disconnect the auxiliary winding from the motor circuit. Capacitor-run motors will typically keep the auxiliary winding in the circuit by either switching to a different value of capacitance under running conditions or by excluding a switch entirely. For the sake of simplicity, this section will deal with an induction-run motor, which could be a split-phase or cap-start motor. Because this section deals with the main winding alone under running conditions, the distinction does not matter.

Equivalent Circuit for the Single-phase Motor

Common to the polyphase equivalent circuit, the equivalent circuit for the single-phase motor includes a transformation by division of the secondary impedances by slip s. This circuit is shown below.

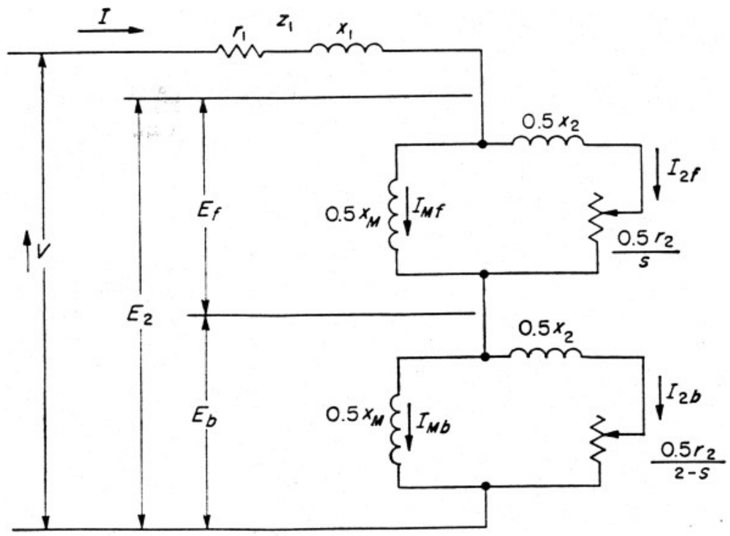

Figure 7, the single-phase equivalent circuit model[3].

Upon inspection there are a few notable differences. Paralleled secondary circuits are shown for the forward and backward fields in relation to the rotor. Neither circuit has a resistor simulating iron losses, which typically makes a negligible difference in single-phase calculations.

The forward revolving field is by definition the field that revolves in the same direction of the rotor at synchronous speed. The impedances for its secondary circuit are thus all divided by slip s. The backward revolving field moves in the opposite direction of the rotor at synchronous speed, so that at any slip s, the relative motion between the rotor and backward field is 2 – s. Each of the impedances of the backward field's secondary are seen divided by this value. The circuit equation for the primary may be observed as

$$V = Ir_1 + jIx_1 + E_f + E_b = Iz_1 + E_2 \qquad (9)$$

Whereas the equations for the secondary, in terms of currents, re

$$I_f = \frac{E_f}{0.5(\frac{r_2}{s} + jx_2)} \qquad (10)$$

$$I_b = \frac{E_b}{0.5(\frac{r_2}{(2-s)} + jx_2)} \qquad (11)$$

It was shown for the polyphase motor that the torque developed was proportional to the secondary I^2r losses. Conductor losses are present in both forward and backward circuits under running conditions. The forward field torque contributes to using motor torque, while backward field torque pulls in the opposite direction, acting as a mechanical drag. The net developed torque is given by

$$T = I^2(R_f - R_b) \qquad (12)$$

where R_f and R_b are the total resistances for the forward and backward secondary circuits, respectively.

Combined Winding Performance Calculations

The previous section dealt with the operation of a single-phase motor when only the main winding is in the circuit. However, most single-phase motors use a starting winding to bring the motor up to speed. A calculation of the performance of the combined windings gives the designer the entire speed-torque curve and helps to determine if the torque will reach the desired switch operating point.

Additional Revolving Fields

With the auxiliary winding in the motor circuit, a total of four revolving fields will be set up; A forward field and a backward field for the main winding, and a forward field and a backward field for the auxiliary. Each field induces a voltage in both of the windings, where voltages induced by fields of the opposite winding are either proportional or inversely proportional to the turns-ratio a:

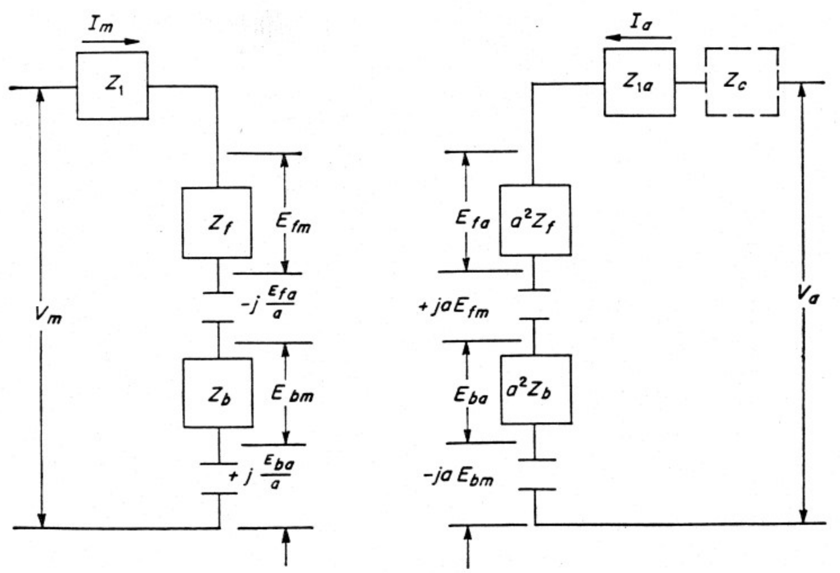

Figure 8, primary windings of the single-phase motor, given by Veinott[3].

This a-ratio is the effective turns of the auxiliary winding to the effective turns of the main. The main winding is shown to the left, the auxiliary to the right. When representing a capacitor motor, an additional reactance Z_c will be in series with the starting winding. The resultant forward wave is the addition of both the forward fields, and it can be shown that this resultant forward wave can be set up by a fictitious current flowing in the main winding, (I_m – jaI_a). Similarly, the backward can be said to be set up by (I_m + jaI_a) flowing in the main winding.

Following the same process discussed before, the forward and backward torques may be by calculated as

$$T_f = (I_m - jaI_a)^2 R_f \qquad (13)$$

$$T_b = (I_m + jaI_a)^2 R_f \qquad (14)$$

The difference is then the developed torque of the combined windings:

$$T = T_f - T_b \qquad (13)$$

Summary

The discussion above has encompassed the general torque characteristics of the single-phase motor, both with and without a starting winding. The revolving-field theory is a time-tested method for its analysis. Similar methods may be used in its place, as shown by Veinott[3].

It will be shown in the following Python implementation of the revolving-field model that other crucial data can be analyzed such as: losses, efficiency, torque pulsations, and the voltage across the capacitor when present.

Scripting the Equivalent Circuits

Python Implementation of the Polyphase Model

The following Python function was developed from a calculation given by Veinott[3]. It consists of a step-by-step method for the prediction of polyphase induction motor performance.

V = volts per phase	= 127
m = no. of phases	= 3
X = s.c. reactance (per ph.)	= 5.0
X_0 = o.c. reactance (per ph.)	= 98.0
$K_p = \sqrt{\dfrac{X_0 - X}{X_0}}$	= 0.970
Friction and windage (watts)	= 20
I_0 = no load amps = V/X_0	= 1.297
Core loss (total)	= 37
$r_M = \dfrac{\text{core loss}}{m \times I_0^2}$	= 7.32
r_1 = pri. res. per phase	= 2.40
$F_2 = r_2$	= 1.55
$F_3 = r_2 \dfrac{r_1 + r_M}{X_0}$	= 0.1537
$F_1 = X - \dfrac{r_M}{X_0} r_1$	= 5.82
$F_4 = r_1 + \dfrac{r_M}{X_0} X$	= 2.85
$F_5 = I_0 r_M$	= 9.49
$F_6 = I_0 r_2$	= 2.01
$F_7 = V K_p$	= 123.2
Stg. torque = $\dfrac{112.7}{\text{syn. rpm}} \times (18)_{s=1}$ (in oz-ft)	
Slip at max. torque = $s_m = \dfrac{r_2}{\sqrt{r_1^2 + X^2}}$ = 0.240	

Item		Full load	Stg. T.	Max. T.
1	Slip = s (assume)	0.0297	1.00	0.24 *
2	$1/s$	33.7	1.00	4.17 *
3	$1 - s$	0.970	0.00	0.76 *
4	F_1	5.82	5.82	5.82 *
5	$F_3 \times (2)$	5.17	0.15	0.64 *
6	$U = (5) - (4)$	−0.65	−5.67	−5.18 *
7	$F_4 =$	2.85	2.85	2.85 *
8	$F_2 \times (2)$	52.3	1.55	6.46 *
9	$W = (7) + (8)$	55.2	4.40	9.31 *
10	$\sqrt{U^2 + W^2}$	55.2	7.17	10.66 *
11	F_5	9.49		
12	$F_6 \times (2)$	67.75		
13	(11) + (12)	77.24		
14	$\sqrt{(13)^2 + V^2}$	148.6		
15	$I = (14) \div (10)$	2.69		
16	$I_2 = F_7 \div (10)$	2.235	17.20	11.56 *
17	Pri. loss = $I^2 \times r_1 \times m$	52		
18	Sec. loss = $I_2^2 \times r_2 \times m$	23.4	1375	620 *
19	Core loss	37		
20	Sec. output = (18) × (2) × (3)	766		1961 *
21	Input = (17) + (18) + (19) + (20)	878		
22	Friction and windage	20		15 *
23	Output = (20) − (22)	746		1946 *
24	Rpm = (3) × syn. rpm	1746		1369 *
25	Torque = 112.7 × (23) ÷ (24)	48.1	86	160 *
26	Eff. = (23) ÷ (21) × 100	84.9		
27	P.F. = (21) × 100/$VI.m$	85.7		
28	% full load	100		

* Terms to get output and torque.

Figure 9, main-winding performance calculation under running conditions[3].

Useful and remarkable as this sheet is, it was used in the era of the slide-rule and was typically meant to determine the characteristics at three load points. Today, we are much more fortunate with modern calculating tools. The definition of the function for a poly-phase model is shown below.

```python
def poly_circuit(volts, ph, x1, x2, xM, r1, r2, rfe, f_w, sync):
    '''
    This function computes the performance of an induction motor given its equivalent
    circuit parameters aka motor constants.
    Args:
        volts: Phase-voltage value [volts]
        ph:    No. of phases
        x1:    Primary leakage reactance [ohms]
        x2:    Secondary leakage reactance [ohms]
        xM:    Mutual reactance [ohms]
        r1:    Primary resistance [ohms]
        r2:    Secondary resistance [ohms]
        rfe:   Resistance for iron losses [ohms]
        f_w:   Friction and windage loss [watts]
        sync:  Synchronous speed of machine [rpm]
    Returns:
        perf:  Dictionary containing performance over range of operating points
    '''
    import numpy as np

    # series core-loss resistance
    rM = xM**2 / (rfe * (1 + (xM/rfe)**2))

    # F-CONSTANTS
    F1 = x1 + x2 * xM / (xM + x2) - r1 * rM / (xM + x2)
    F2 = r2 * (xM + x1) / (xM + x2)
    F3 = r2 * (r1 + rM) / (xM + x2)
    F4 = rM * (x1 + x2) / (xM + x2) + r1
    F5 = volts * rM / (xM + x2)
    F6 = volts * r2 / (xM + x2)
    F7 = volts * np.sqrt(rM**2 + xM**2) / (xM + x2)
    A10 = volts * x2 / (xM + x2)

    # points calculated: no-load plus operation
    slip_points = [0.0001, 0.0005, 0.001, 0.005] + [abs(round(i, 2)) for i in np.arange(0.01, 1.01, 0.01)]

    # dictionary to hold expected performance data
    perf = {'slip': slip_points,
            'current': [],
            'pri_loss': [],
            'sec_loss': [],
            'core_loss': [],
            'input': [],
            'torque': [],
            'eff': [],
            'pf': []}

    for s in slip_points:

        part1 = np.sqrt((F5 + F6 / s)**2 + volts**2)
        part2 = np.sqrt((F3 / s - F1)**2 + (F2 / s + F4)**2)

        I = part1 / part2 # primary current
        I2 = F7 / part2 # secondary current
        IM = np.sqrt(((F6/s)**2 + A10**2) / part2**2) # magnetizing current

        pri_loss = I**2 * r1 * ph
        sec_loss = I2**2 * r2 * ph
        core_loss = IM**2 * rM * ph

        if s != 1:
            sec_output = sec_loss * (1 - s) / s
            if sec_output > f_w:
                output = sec_output - f_w
            else: output = 0
        else:
            sec_output = 0
            output = 0
```

```python
            input_ = pri_loss + sec_loss + core_loss + sec_output

            torque = 7.04 * (sec_loss - f_w) / (s * sync)

            eff = output / input_ * 100

            pf = input_ / (volts * I * ph) * 100

            # add perf for this speed in results dict
            perf['current'].append(I)
            perf['pri_loss'].append(pri_loss)
            perf['sec_loss'].append(sec_loss)
            perf['core_loss'].append(core_loss)
            perf['input'].append(input_)
            perf['torque'].append(torque)
            perf['eff'].append(eff)
            perf['pf'].append(pf)

        return perf
```
[2] ✓ 0.0s

Upon inspection of this function, it is seen that the entire range of slip is calculated for all performance characteristics. Plotting the characteristics for the example in the calculation sheet verifies the theory of no-load, locked-rotor, and running conditions previously discussed:

```python
import matplotlib.pyplot as plt
import numpy as np

# call function to create model
motor_data = poly_circuit(
                            volts = 460/np.sqrt(3),
                            ph = 3,
                            x1 = 10.4944,
                            x2 = 7.1384,
                            xM = 229.4160,
                            r1 = 8.8667,
                            r2 = 2.9103,
                            rfe = 15771.7881,
                            f_w = 15,
                            sync = 1800
                        )


# plot perf results
slip = np.array(motor_data['slip'])
torq = np.array(motor_data['torque'])
watts = np.array(motor_data['input'])
amps = np.array(motor_data['current'])
eff = np.array(motor_data['eff'])
pf = np.array(motor_data['pf'])

fig, ax = plt.subplots()
fig.subplots_adjust(right=0.75)

twin1 = ax.twinx()
twin2 = ax.twinx()
twin3 = ax.twinx()
twin4 = ax.twinx()

twin2.spines.right.set_position(("axes", 1.2))
twin3.spines.right.set_position(("axes", 1.4))
twin4.spines.right.set_position(("axes", 1.6))

p1, = ax.plot(slip, torq, "C0", label="Torque, lb-ft")
p2, = twin1.plot(slip, watts, "C1", label="Line Watts")
p3, = twin2.plot(slip, amps, "C2", label="Line Amps")
p4, = twin3.plot(slip, eff, "C3", label="Efficiency")
p5, = twin3.plot(slip, pf, "C4", label="Power Factor")

ax.set(ylim=(0, max(torq)+3), xlabel="Slip", ylabel="Torque, oz-ft")
ax.set_xlim(1, 0)
twin1.set(ylim=(0, max(watts)+500), ylabel="Line Watts")
twin2.set(ylim=(1,  max(amps)+15), ylabel="Line Amps")
twin3.set(ylim=(0, 100), ylabel="Efficiency")
twin4.set(ylim=(0, 100), ylabel="Power Factor")

ax.yaxis.label.set_color(p1.get_color())
twin1.yaxis.label.set_color(p2.get_color())
twin2.yaxis.label.set_color(p3.get_color())
twin3.yaxis.label.set_color(p4.get_color())
twin4.yaxis.label.set_color(p5.get_color())

ax.tick_params(axis='y', colors=p1.get_color())
twin1.tick_params(axis='y', colors=p2.get_color())
twin2.tick_params(axis='y', colors=p3.get_color())
twin3.tick_params(axis='y', colors=p4.get_color())
twin4.tick_params(axis='y', colors=p5.get_color())

ax.grid()
ax.set_title('Performance data')

plt.show()
```

[3] ✓ 0.3s

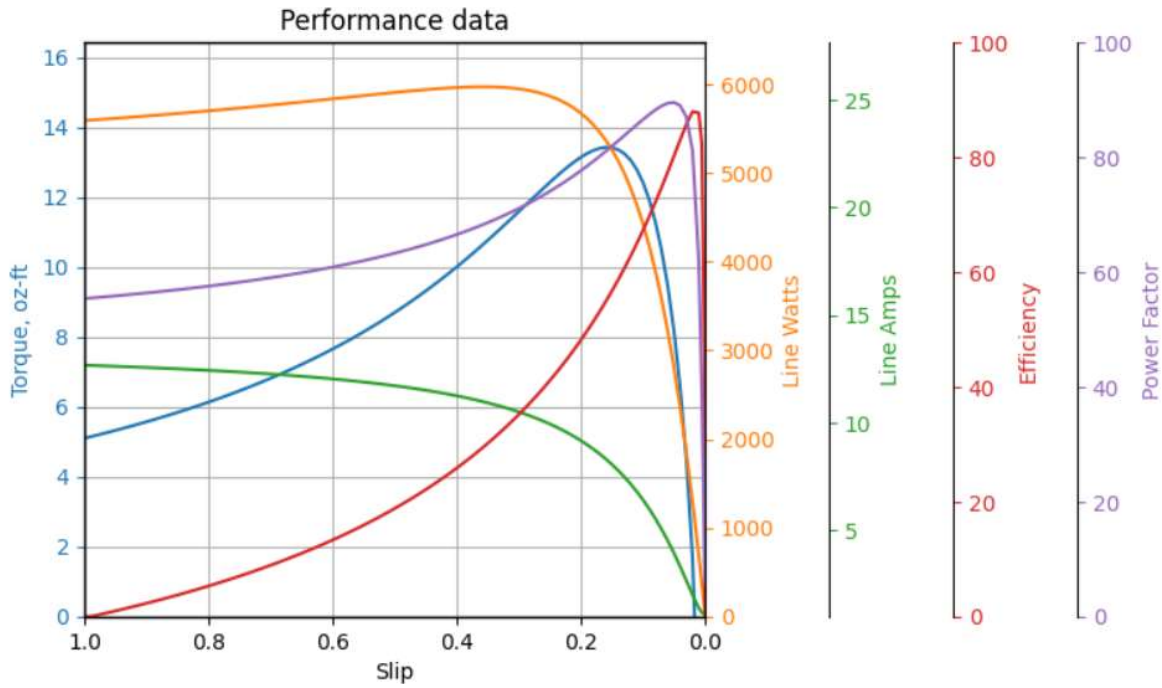

Figure 10, performance calculations from the implementation of the polyphase equivalent circuit.

By a similar method, the losses returned from the function can be analyzed as well:

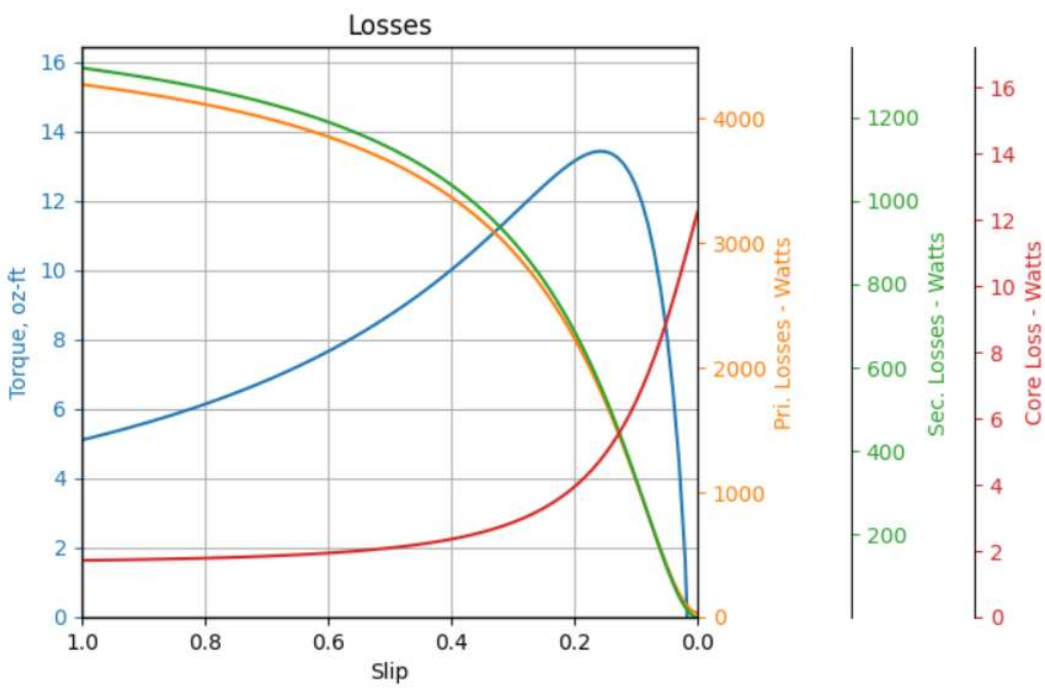

Figure 11, loss calculations from the equivalent circuit model.

Python Implementation of the Revolving-field Theory

The calculating method used here for the revolving-field theory of the single-phase motor was originally developed by W.J. Morrill in 1929. Veinott refactors his method into the following sheets.

For the main winding only:

Given:	
r_1 hot	3.80
r_2 hot	4.65
X (s.c. react.)	8.3
X_0 (o.c. react.)	110
K_p	.964
K_r	.929
Fe loss (ff)	10.0
Fe loss (hf)	9.0
Fe loss (total)	19.0
Fr. & Wind.	10.0
Hp rating	⅙
Volts	110
Cycles	60
Poles	4

Calculate:			
$x_1 = X/(1 + K_p)$	4.23	$M_3 = 0.5K_pX_0$	53.0
$M_1 = 0.5K_rr_2$	2.16	$M_4 = 0.5K_px_1$	2.04
$M_2 = r_2/X_0$	.0423		

							No load	Break-down torque
1	s	.044						.19
2	$2 - s$	1.956						1.81
3	M_1/s	49.1						11.37
4	$M_1/(2 - s)$	1.104						1.193
5	$(M_2/s)^2$	.924						.0496
6	$[M_2/(2 - s)]^2$	.000467						.00055
7	1.0 + (5)	1.924						1.0496
8	1.0 + (6)	1.0005						1.00055
9	R_f = (3)/(7)	25.5						10.83
10	R_b = (4)/(8)	1.103						1.192
11	r_1 =	3.8						3.80
12	$R = R_f + R_b + r_1$	30.40						15.82
13	M_3 × (5)	49.0						2.62
14	M_3 × (6)	.025						.0292
15	M_4	2.04						2.04
16	(13) + (15)	51.04						4.66
17	(14) + (15)	2.065						2.07
18	X_f = (16)/(7)	26.5						4.44
19	X_b = (17)/(8)	2.06						2.07
20	x_1	4.23						4.23
21	$X = X_f + X_b + x_1$	32.8						10.74
22	$Z = \sqrt{(12)^2 + (21)^2}$	44.7						19.12
23	$I = V/(22)$	2.46						5.75
24	I (corrected) *	2.52					1.860	
25	$R_f - R_b$	24.4						9.638
26	Pri. loss = $(24)^2 \times r_1$	24.1					13.1	
27	Sec. loss (f) = $(23)^2 sR_f$	6.8	19.9				0.0	
28	" " (b) = $(23)^2(2 - s)R_b$	13.1					7.5	
29	Fe loss (ff)	10.0					10.0	
30	$(23)^2 \times (25) \times (1 - s)$	141.2					19.0	258.1
31	Input = 26 to 30 inc.	195.2					49.6	
32	Fe loss (hf) + F & W	19.0						19.0
33	Output = (30) − (32)	122.1						239.1
34	Rpm = (1 − s) syn. rpm	1721						1458
35	Torq. = 112.7(33)/(34)	8.00						18.5
36	Eff. = (33)/(31)	62.6						
37	P.F. = (31)/(24) V	70.4						
38	App. Eff. = (36) × (37)	44.1						
39	% Full load	98.2						

* I (corrected)

$$= (23) + \frac{\text{Fe (ff)}}{V} \times \frac{(12)}{(22)}$$

No-load Calcs.	
24	$I_0 = 2V/(X_0 + X)$
27	Sec. loss (f) = 0
28	Sec. loss (b) = $I_0^2 M_1$

Figure 12, performance calculation for the main-winding under running conditions[3].

For the combined winding performance:

V_m	115	V_a	115
X_0	20.55	$K_\epsilon = V_a/V_m$	1.00
X_m	1.52	a	1.25
Hp	.75	a^2	1.563
K_r	.926	Syn. rpm	1800
K_p	.960	Fe loss	94.7
X_a	2.37	F & W	13.3
r_2	.854	Fe + F & W	108.0

Z_1	r_1 (hot) = .695		$x_1 = \dfrac{X_m}{1+K_p} = .775$	
Z_{1a}	r_{1a} (hot) = 2.70		$x_{1a} = \dfrac{X_a}{1+K_p} = 1.21$	
Z_c	R_c (hot) = 0.46		$X_c = -5.82$	
$Z_{1a} + Z_c$	$r_{1a} + R_c = 3.16$		$x_{1a} + X_c = -4.61$	
$M_1 = 0.5K_r r_2 =$		.395	$M_3 = 0.5K_p X_0 =$	9.86
$M_2 = r_2/X_0 =$		.0415	$M_4 = 0.5K_p x_1 =$	.372

			r-term	j-term	r-term	j-term	r-term	j-term	r-term	j-term
1	s = slip	M_4	.20	.372						
2	$(M_2/s)^2$	$M_3(M_2/s)^2$	.04306	.4245						
3	M_1/s	(1) + (2)	1.9750	.7965						
4	$(M_2/s)^2 + 1$		1.04306							
5	$2 - s$	M_4	1.800	.3720						
6	$(M_2/2-s)^2$	$M_3(M_2/2-s)^2$	.00053	.0052						
7	$M_1/2 - s$	(5) + (6)	.2194	.3772						
8	$(M_2/2 - s)^2 + 1$		1.00053							
9	$R_f = (3)/(4)$	$X_f = (3)/(4)$	1.8935	.7636						
10	$R_b = (7)/(8)$	$X_b = (7)/(8)$	.2193	.3770						
11	(9) − (10)		1.6742	.3866						
12	$a \times$ (11)		2.093	.4833						
13	Z_1		.695	.775						
14	(9) + (10)		2.1128	1.1406						
15	$a^2 \times$ (14)		3.3023	1.7828						
16	$Z_{1a} + Z_c$		3.160	−4.61						
17	Z_T = (13) + (14)		2.8078	1.9156						
18	Z_{Ta} = (15) + (16)		6.4623	−2.8272						
19	$Z_T \times Z_{Ta}$ (Vector)		23.561	4.441						
20	[Vector (12)]^2		4.147	2.023						
21	(19) − (20)		19.414	2.417						
22	V_m/(21)		5.833	−.7262						
23	(18)		6.4623	−2.8272						
24	$jK_\epsilon \times$ (12) (Vector)		−.4833	2.093						
25	(23) + (24)		5.9790	−.7342						
26	$K_\epsilon \times$ (17)		2.8078	1.9156						
27	$-j \times$ (12) (Vector)		.4833	−2.0930						
28	(26) + (27)		3.2911	−.1774						
29	$\bar{I}_m$ = (25)(22) (Vector)		A = 34.342	B = −8.625	A =	B =	A =	B =	A =	B =
30	$\bar{I}_a$ = (28)(22) (Vector)		g = 19.068	h = −3.425	g =	h =	g =	h =	g =	h =
31	$\bar{I}_L = \bar{I}_m + \bar{I}_a$		53.410	−12.050						

(continued on next page)

32	Ah	−117.62							
33	Bg	−164.46							
34	$Ah - Bg$	+46.84							
35	$I_m = \sqrt{A^2 + B^2}$	35.41							
36	$I_a = \sqrt{g^2 + h^2}$	19.37							
37	Line Amps	54.75							
38	Watts, Mn. Ph. = $V_m A$	3946.8							
39	Watts, Aux. Ph. = $V_a g$	2192.8							
40	Line Watts = (38) + (39)	6139.6							
41	I_m^2	1253.76							
42	$(aI_a)^2$	586.63							
43	(41) + (42)	1840.39							
44	$2a \times (34)$	117.10							
45	(43) + (44)	1957.49							
46	(43) − (44)	1723.29							
47	(45) × R_f	3706.5							
48	(46) × R_b	377.9							
49	(47) − (48)	3328.6							
50	$1 - s$	.800							
51	(49) × (50)	2662.9							
52	(Fe + F & W)$(1 - s)$	86.4							
53	Output Watts = (51) − (52)	2576.5							
54	Efficiency = (53)/(40)	.4197							
55	P.F. = (40)/(37) × V_m	.9755							
56	Rpm = (50) × syn. rpm	1440.0							
57	Torque, oz-ft	201.6							
58	Sec. $I^2R(f)$ = (47) s	741.3							
59	Sec. $I^2R(b)$ = (48)$(2 - s)$	680.2							
60	Mn. Wdg. Cu. loss = $r_{1m} I_m^2$	871.4							
61	Aux. Wdg. Cu. loss = $r_{1a} I_a^2$	1013.4							
62	Capacitor loss = $R_c I_a^2$	172.6							
63	(Fe + F & W)$(1 - s)$	86.4							
64	Total losses	3565.3							
65	Output + losses	6141.8							
66	Cap. Volts = $I_a \sqrt{R_c^2 + X_c^2}$	113.02							
67	Aux. wdg. volts = $V - (30) Z_c$	169.07							
68	$T_p = \sqrt{[(11r)^2 + (11j)^2]\,(45)(46)}$	3158							
69									
70									
71									

Figure 13, combined winding performance calculation, as given by Veinott[3].

As mentioned earlier, one of the most daunting tasks in designing a capacitor motor is removal of the starting winding so that (1) the capacitor volts do not exponentially grow and destroy the capacitor and (2) the thinner starting winding does not burn out. Typical switch operating points are about 75% of synchronous speed. The figure below shows the importance of the opening of the switch.

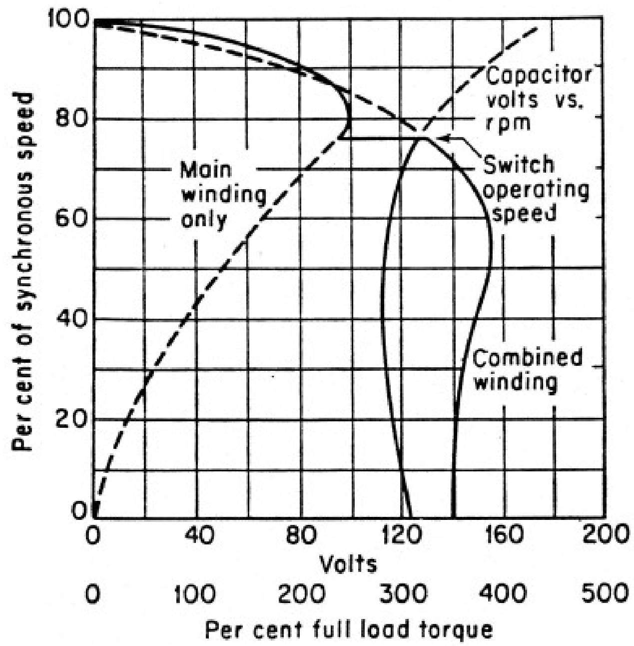

Figure 14, characteristic torque operation for a single-phase induction-run motor[3].

It is noted that for cap-start motors, there is a point at which the main winding alone performs better than the combined winding performance; another reason for the opening switch.

Python functions are implemented separately for the main-only and combined-winding circuits. The function for the main winding is shown below.

```python
def get_main_perf(r1, r2, X, Xo, Kr, feff, fehf, f_w, hp, volts, hertz, poles, sync):
    '''
    This function computes the performance of an induction motor given its equivalent
    circuit parameters aka motor constants.
    Args:
        r1:    Primary resistance [ohms]
        r2:    Secondary resistance [ohms]
        X:     Short-circuit reactance [ohms]
        Xo:    Open-circuit reactance [ohms]
        Kr:    Reactance Constant
        feff:  Fundamental frequency surface losses [ohms]
        fehf:  High frequency surface losses [ohms]
        f_w:   Friction and windage loss [watts]
        hp:    Output [horsepower]
        volts: Phase-voltage value [volts]
        hertz: Operating frequency
        poles: No. of poles in machine
        sync:  Synchronous speed of machine [rpm]
    Returns:
        rpms:    Speed range [RPM]
        torques: Torque range [lb-ft]
    '''
    import numpy as np

    Kp = round(np.sqrt((Xo - X) / Xo), 3)

    # Calculate:
    x1 = round(X / (1 + Kp), 3)
    M1 = round(0.5 * Kr * r2, 2)
    M2 = round(r2 / Xo, 4)
    M3 = round(0.5 * Kp * Xo, 0)
    M4 = round(0.5 * Kp * x1, 2)

    slip_points = [0.0001, 0.0005, 0.001, 0.005] + [abs(round(i, 2)) for i in np.arange(0.01, 1.0, 0.01)]
    torques = []
    rpms = []

    for s in slip_points:
        item1 = s
        item2 = 2 - item1
        item3 = M1 / item1
        item4 = M1 / item2

        item5 = (M2 / item1)**2
        item6 = (M2 / item2)**2
        item7 = 1 + item5
        item8 = 1 + item6
        item9 = item3 / item7
        item10 = item4 / item8
        item11 = r1
        item12 = item9 + item10 + item11
        item13 = M3 * item5
        item14 = M3 * item6
        item15 = M4
        item16 = item13 + item15
        item17 = item14 + item15
        item18 = item16 / item7
        item19 = item17 / item8
        item20 = x1
        item21 = item18 + item19 + item20
        item22 = np.sqrt(item12**2 + item21**2)
        item23 = volts / item22
        item24 = item23 + feff / volts * item12 / item22
        item25 = item9 - item10
        item26 = round(item24**2 * r1, 3) # PRIMARY LOSS
        item27 = round(item23**2 * item1 * item9, 3) # SEC LOSS FORW
        item28 = round(item23**2 * item2 * item10, 3) # SEC LOSS BACK
        item29 = feff
        item30 = round(item23**2 * item25 * (1 - item1), 3) # developed output
        item31 = round(item26 + item27 + item28 + item29 + item30, 3) # INPUT WATTS
        item32 = fehf + f_w
        item33 = item30 - item32 # OUTPUT SYNC WATTS
        item34 = round((1 - item1) * sync, 0) # RPM
        item35 = round(112.7 * item33 / item34, 3) # TORQUE
        item36 = round(item33 / item31 * 100, 1)  # EFFICIENCY
        item37 = round(item31 / (item24 * volts) * 100, 1)  # PF

        torques.append(item35)
        rpms.append(item34)

    return rpms, torques
```

[33]

The more involved combined-winding function is shown as

```python
def get_combined_perf(Vm, Xo, Xm, Hp, Kr, Kp, Xa, r2, Va, a, sync, fe_loss, f_w, r1, r1a, Rc, Xc):
    '''
    This function computes the performance of an induction motor given its equivalent
    circuit parameters aka motor constants.
    Args:
        Vm:      Main-winding voltage [volts]
        Xo:      Open-circuit reactance [ohms]
        Xm:      Magnetizing reactance [ohms]
        Kr:      Reactance Constant
        Kp:      Reactance Constant
        r2:      Secondary resistance [ohms]
        Va:      Aux-winding voltage [volts]
        a:       Turns-ratio
        sync:    Synchronous speed of machine [rpm]
        fe_loss: Iron losses [ohms]
        f_w:     Friction and windage loss [watts]
        r1:      Main-winding resistance [ohms]
        r1a:     Aux-winding resistance [ohms]
        Rc:      Capaction resistance [ohms]
        Xc:      Capcitor reactance [ohms]
    Returns:
        rpms:      Speed range [RPM]
        torques:   Torque range [lb-ft]
        cap_volts: Capacitor voltage range [volts]
    '''
        import numpy as np

        rpms = []
        torques = []
        cap_volts = []
        # Z1
        x1 = Xm / (1 + Kp)
        Z1 = complex(r1, x1)
        # Z1a
        x1a = Xa / (1 + Kp)
        Z1a = complex(r1a, x1a)
        # Zc
        Zc = complex(Rc, Xc)

        M1 = 0.5 * Kr * r2
        M2 = r2 / Xo
        M3 = 0.5 * Kp * Xo
        M4 = 0.5 * Kp * x1
        Ke = Va /Vm
        # points calculated: no-load plus operation
        slip_points = [0.0001, 0.0005, 0.001, 0.005] + [abs(round(i, 2)) for i in np.arange(0.01, 1.0, 0.01)]

        # dictionary to hold expected performance data
        perf = {'slip': slip_points,
                'line_amps': [],
                'line_watts': [],
                'torque': [],
                'eff': [],
                'pf': [],
                'main_amps': [],
                'aux_amps': [],}

        for s in slip_points:
                item_1 = complex(s, M4)
                item_2 = complex((M2/s)**2, M3*(M2/s)**2)
                item_3 = complex(M1/s, np.imag(item_1 + item_2))
                item_4 = (M2/s)**2 + 1
                item_5 = complex(2-s, M4)
                item_6 = complex((M2/(2-s))**2, M3*(M2/(2-s))**2)
                item_7 = complex(M1/(2-s), np.imag(item_5 + item_6))
                item_8 = (M2/(2-s))**2 + 1

                # item 9
                Zf = complex(np.real(item_3/item_4), np.imag(item_3/item_4))
                # item 10
                Zb = complex(np.real(item_7/item_8), np.imag(item_7/item_8))

                item_11 = Zf - Zb
                item_12 = a * item_11
                # item 13 is Z1
                item_14 = Zf + Zb
                item_15 = a**2 * item_14
                item_16 = Z1a + Zc
                item_17 = ZT = Z1 + item_14
                item_18 = ZTa = item_15 + item_16
                item_19 = ZT * ZTa
                item_20 = item_12**2
                item_21 = item_19 - item_20
                item_22 = Vm / item_21
```

```
            # item 23 is item 18
            item_24 = item_12 * Ke * 1j
            item_25 = item_18 + item_24
            item_26 = Ke * item_17
            item_27 = -1j * item_12
            item_28 = item_26 + item_27
            item_29 = Im = item_25 * item_22
            item_30 = Ia = item_28 * item_22
            item_31 = IL = Im + Ia

            Ah = np.real(Im) * np.imag(Ia)
            Bg = np.imag(Im) * np.real(Ia)
            item_34 = Ah - Bg
            item_34

            Im_ = np.sqrt(np.real(Im)**2 + np.imag(Ia)**2)
            Ia_ = np.sqrt(np.real(Ia)**2 + np.imag(Ia)**2)
            line_amps  = Im_ + Ia_
            watts_main = Vm*np.real(Im)
            watts_aux  = Va * np.real(Ia)
            line_watts = watts_main + watts_aux

            item_41 = Im_**2
            item_42 = (a*Ia_)**2
            item_43 = item_41 + item_42
            item_44 = 2*a * item_34
            item_45 = item_43 + item_44
            item_46 = item_43 - item_44
            item_47 = item_45 * np.real(item_3/item_4)
            item_48 = item_46 * np.real(item_7/item_8)
            item_49 = item_47 - item_48
            item_51 = item_49 * (1-s)
            item_52 = (fe_loss + f_w) * (1-s)

            output_watts = item_51 - item_52
            efficiency = output_watts / line_watts
            power_factor = line_watts / (line_amps * Vm)
            rpm = (1-s) * sync

            torque = 112.7 / rpm * output_watts
            sec_I2R_f = item_47 * s
            sec_I2R_b = item_48 * (2-s)

            main_cu_loss = r1 * Im_**2
            aux_cu_loss = r1a * Ia_**2
            cap_loss = Rc * Ia_**2
            total_loss = main_cu_loss + aux_cu_loss + cap_loss + sec_I2R_b + sec_I2R_f + item_52

            input_ = output_watts + total_loss
            cap_volt = np.sqrt(Rc**2 + Xc**2) * Ia_
            aux_volts = abs(Va - (item_30*Zc))

            rpms.append(rpm)
            torques.append(torque)
            cap_volts.append(cap_volt)

    return rpms, torques, cap_volts
```

[ ]

Calling both these functions for the example given in the combined-winding performance calculation sheet and plotting the torques for each, along with the volts across the capacitor, gives a similar figure to previous speed-torque curve shown.

```
# combined winding
rpms_comb, torques_comb, cap_volts = get_combined_perf(Vm=115, Xo=20.55, Xm=1.52, Hp=0.75, Kr=0.926, Kp=0.96, Xa=2.37, r2=0.854,
                                        Va=115, a=1.25, sync=1800, fe_loss=94.7, f_w=13.3, r1=0.695, r1a=2.7, Rc=0.46, Xc=-5.82)

# main only
rpms_main, torques_main = get_main_perf(r1=0.695, r2=0.854, X=1.32, Xo=12.55, Kr=0.926, feff=20, fehf=19, f_w=13.3, hp=0.75,
                                        volts=115, hertz=60, poles=4, sync=1800)

rpms_comb = np.array(rpms_comb)
torques_comb = np.array(torques_comb)

rpms_main = np.array(rpms_main)
torques_main = np.array(torques_main)

fig, ax = plt.subplots()
fig.subplots_adjust(right=0.75)
twin1 = ax.twinx()

p1, p2 = ax.plot(rpms_comb, torques_comb, "b", rpms_main, torques_main, "r", label="Torque, oz-ft")
p3, = twin1.plot(rpms_comb, cap_volts, "g", label="Capacitor voltage")

ax.set(ylim=(0, max(torques_comb)+45), xlabel="RPM", ylabel="Torque, oz-ft")
ax.set_xlim(1, 1800)
twin1.set(ylim=(0, max(cap_volts)+5), ylabel="Capacitor voltage")

ax.grid()
ax.set_title('Capacitor Motor Performance')
ax.legend(['Combined Performance', 'Main winding only'], loc='upper left')
twin1.legend(['Cap Volts'], loc='upper right')
plt.show()
```

[71] ✓ 0.1s

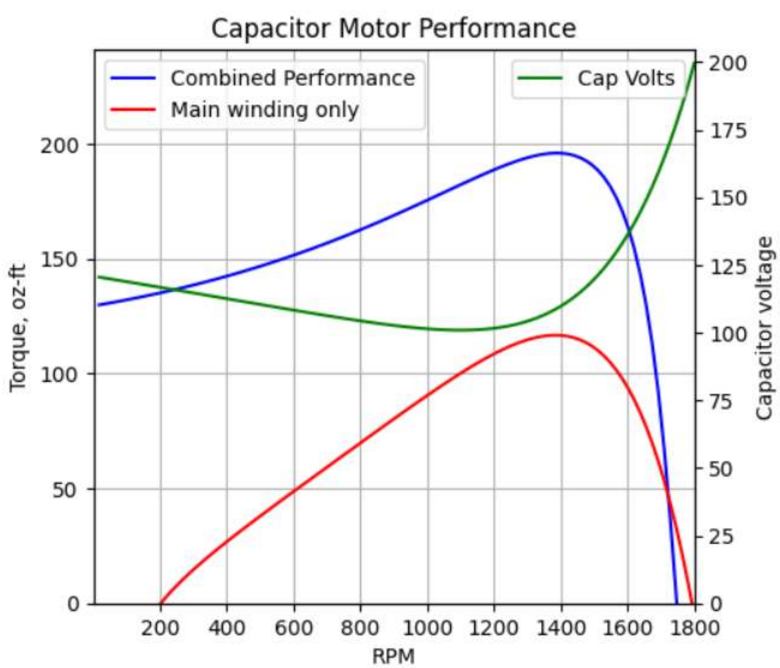

Figure 15, combined- and main-winding torques with capacitor volts.

This can be a very power tool in analyzing required torques in single-phase motors.

Conclusion

It seems as though the majority of literature on the design of induction motors printed since 2000 has been predominantly academic; they are concerned with notoriously difficult field equations and finite element modeling. Though the finite element method is useful for refinement, it is not used in practical design work. Neglected as well in FEA are costs of materials, ratio methods, and harmonization with standards. Equivalent circuits remain dominant in industry, even in thermal calculations. The clever physical models have been in use well over a hundred years, and by the author's experience in industry they are here to stay a hundred more.

References

[1] D. Meeker. Induction Motor Example, 2004 Available: http://www.femm.info/wiki/InductionMotorExample

[2] D. Dolinar *et al.,* "Calculation of two-axis induction motor model parameters using finite elements," *IEEE Transactions on Energy Conversion,* June 1997.

[3] C. G. Veinott, Theory and Design of Small Induction Motors, McGraw-Hill, New York, 1959

[4] C. I. Hubert, Electric Machines: Theory, Operation, Applications, Adjustment, and Control, Merrill, 2001