Optimizacion Multiobjetivo

Proyecto Final

Mauricio Rodriguez Calvo
Maestría Académica en Ingeniería Eléctrica
Universidad de Costa Rica
San José, Costa Rica
mauricio.rodriguezcalvo@ucr.ac.cr

***Abstract*—En este documento se estudia el sistema térmico de un motor de corriente continua, basado en la red generalizada propuesta por [7]; para llevar a cabo el análisis de variables como el calor en el núcleo del motor, la resistencia térmica y la capacitancia térmica del motor. Estas variables son utilizadas en dos funciones objetivo para realizar una optimización multiobjetivo del sistema y encontrar el respectivo frente de Pareto y conjunto de Pareto con los valores óptimos alcanzados por los algoritmos de optimización. Los algoritmos de optimización seleccionados son el evMODAD, el cual es parte de la familia de algoritmos genéticos y la Restricción Normal Normalizada (NNC). (*Abstract*)**

I. DESCRIPCIÓN DE LA APLICACIÓN

Los motores eléctricos representan la principal fuerza motriz para llevar a cabo el movimiento de las diferentes partes móviles de un robot [1], en donde se requiere una alta eficiencia energética para poder mover objetos de todo tipo y tamaño, así como también para vencer el peso y la inercia de los mismos componentes móviles del robot. Los motores eléctricos son muy utilizados para llevar a cabo el movimiento de las partes móviles en robots humanoides, los cuales deben ejecutar tareas diversas dentro y fuera de industrias y hogares. Esta autonomía exige que el robot pueda manejar altas cargas mecánicas en cada uno de los motores que participan en el movimiento completo para llevar a cabo la acción requerida. Dado este hecho, es que los motores eléctricos utilizados en robots humanoides deben poder entregar la mayor cantidad de energía posible, consumiendo una mayor cantidad de corriente en un intervalo de tiempo dado [2].

Uno de los principales problemas que se presentan a la hora de aumentar la eficiencia energética en un motor eléctrica que va a ser utilizado para aplicaciones prácticas en robots humanoides, es la relación velocidad angular & temperatura interna del motor eléctrica; ya que estas, representadas algebraicamente, proporcionan funciones que se oponen entre si y vienen relacionadas por las mismas variables que afectan a ambas de alguna forma.

Métodos de optimización multiobjetivo se han llevado a cabo ampliamente para generar el diseño más óptimo en motores síncronos y de inducción [3],[4], dando como resultado parámetros adecuados de diseño con respecto a las características de cada motor y las aplicaciones prácticas respectivas al trabajo a realizar en el campo.

Muchos de estos estudios han incorporado sistemas de enfriamiento externo a los motores que requieren soportar una mayor carga mecánica externa y que por ende elevan el calor de los motores a la hora de llevar a cabo un trabajo. Estudios de este tipo los podemos ver en [5] donde se utiliza un sistema de capsulado del chasis de dos motores eléctricos para minimizar la cantidad de calor que se generar en el chasis de ambos motores y de esta forma poder entregar una mayor cantidad de corriente y al bobinado de cobre de los motores, incrementando la corriente de funcionamiento del motor y por ende sus revoluciones por minuto. Este encapsulado especial, también lo vemos en [6], en donde se hace circular líquido refrigerante por medio de los conductos del encapsulado que envuelve la parte externa del estator del motor utilizando un sistema de enfriamiento que dentro de sus componentes posee un radiador y un sistema de control de flujo. En esta investigación se propone estudiar el sistema térmico de un motor de corriente continua basado en la red generalizada para generar dos funcione objetivos que se oponen entre si y llevar a cabo una optimización multiopjetivo por medio de los algoritmos EvaMODA y NNC para encontrar su respectivo frente de Pareto y conjunto de Pareto con respecto a los óptimos obtenidos.

II. PLANTEAMIENTO DEL PROBLEMA DE OPTIMIZACIÓN MULTIPOJETIVO

El problema de optimización propuesto representa el estudio del sistema térmico de un motor de corriente continua basado en el circuito térmico en la figura 1 según [8] y la red generalizada; la cual es una manera uniforme de estudiar cualquier sistema dinámico. Este estudio nos va a permitir analizar el modelo de un sistema térmico sencillo que representa el modelo térmico equivalente de un motor de corriente continua, en donde se buscar una función que lo presente de forma equivalente mediante el análisis de su ecuación diferencial para obtener el equivalente a la función en el dominio del tiempo a partir del dominio de la frecuencia.

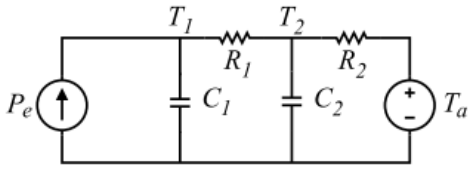

Figura 1: Modelo térmico de un motor de corriente continua, tomado de [8].

Donde T_1,T_2 son la temperatura en el rotor y el estamos respectivamente, C_1,C_2 son las capacitancias térmicas en el rotor y el estator, T_a representa la temperatura ambiente y P_e representa las perdidas óhmicas en el motor. Cabe mencionar que las pérdidas producidas en el proceso de conversión de energía eléctrica a mecánica llevada a cabo por el motor son producto de la fricción mecánica y las perdidas óhmicas.

Una vez estudiado este modelo, se llega a un modelo equivalente que toma en cuenta el flujo de calor entrante a la máquina $q_i(s)$ y que unifique las temperaturas del rotor y estator en una sola temperatura equivalente $T(s)$, junto con una resistencia equivalente y una capacitancia equivalente $C(s)$ e integrando la temperatura ambiente $T_a(s)$. El modelo térmico descrito se muestra en la figura 2.

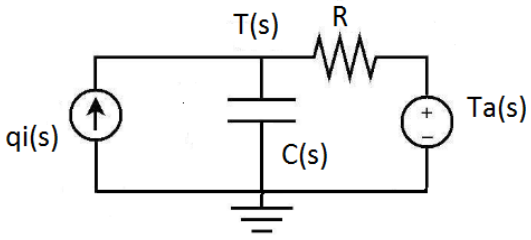

Figura 2: Modelo térmico equivalente del motor de corriente continua.

Teniendo el modelo térmico de la figura 2, se procede a analizar el sistema para encontrar la ecuación diferencial que ayude identificar $T(s)$ en el sistema en el dominio de la frecuencia y luego en el dominio del tiempo.

Sabiendo que:

$$q_i(s) = i_R + i_c \qquad (1)$$

Y que i_R, i_c se pueden expresar de la siguiente manera:

$$i_R(s) = \frac{T(s) - T_a(s)}{R} \qquad (2)$$

$$i_c = \frac{V_c}{1/SC} \qquad (3)$$

Puedo realizar entonces el siguiente cálculo

$$q_i(s) = SC\,T(t) + \frac{1}{R}T(s) - \frac{T_a(s)}{R} \qquad (4)$$

Puedo despejar $T(s)$, la cual queda de la siguiente manera.

$$T(s) = \left(\frac{R}{SRC+1}\right)q_i(s) + \left(\frac{1}{SRC+1}\right)T_a(s) \qquad (5)$$

Una vez calculado $T(s)$ en el dominio de la frecuencia, puedo calcular $T(s)$ en el dominio del tiempo.

$$T(t) = \frac{e^{-\left(\frac{t}{RC}\right)}}{C}q_i(t) + \frac{e^{-\left(\frac{t}{RC}\right)}}{RC}T_a(t) \qquad (6)$$

En la ecuación (6) se expresa la primera función objetivo, teniendo como variables la resistencia térmica, la capacitancia térmica y el flujo de calor entrante al sistema.

Para la segunda función objetivo se va a tomar la ecuación de eficiencia general de un sistema para relacionar la velocidad angular del motor con respecto a la temperatura $T(t)$ que se existe dentro del motor; donde K es una constante definida. Esta viene dada de la siguiente forma.

$$V_m = K\left(\frac{e^{-\left(\frac{t}{RC}\right)}}{C}q_i(t) + \frac{e^{-\left(\frac{t}{RC}\right)}}{RC}T_a(t)\right) \qquad (7)$$

III. SELECCIÓN DEL ALGORITMO DE OPTIMIZACIÓN

A. *Algoritmo evMOGA*

El algoritmo ev-MOGA es parte de los optimizadores multiobjetivo evolutivos y tienen el mismo comportamiento que los algoritmos genéticos en sus componentes de cantidad de población, cantidad de generaciones y un numero de divisiones por dimensión. Estos componentes deben ser variados para obtener precisión a la hora de buscar el conjunto y frente de Pareto y su respectivo frente de Pareto. La cantidad de divisiones es la cantidad de subconjunto que el método utiliza para probar el J1 y J2 a optimizar; es decir; se ejecutan el número de divisiones definidas por cada J y basado en los resultados el método converge a los puntos encontrados.

B. *Algoritmo NNC*

El algoritmo Normalized Normal Constraint (NNC); por sus siglas en inglés; genera un conjunto de soluciones uniformemente espaciadas en un frente de Pareto para problemas de optimización multiobjetivo. Este método le asigna un sistema de mapeo lineal critico a los objetivos;

llevando esto a otorgar una propiedad dinámica, en la que el rendimiento resultante del método es totalmente independiente de las escalas de objetivos de diseño. Los cuales pueden ser realmente difíciles de resolver. Según lo visto por [9] este método debe lograr los siguientes parámetros;

- El método debe ser capaz de generar todas las soluciones de Pareto existentes.
- El método debe ser relativamente fácil de implementar.
- El método debe generar soluciones de Pareto

IV. OPTIMIZACIÓN

La simulación de la optimización se llevó a cabo en Matlab. Utilizando el optimizador genético ev-MOGA y NNC los cuales son optimizadores multiobjetivo. A continuación se muestra el código utilizado para llevar a cabo la optimización. Para el caso de la simulación con el algoritmo ev-MOGA (tabla I) se tomó como ejemplo el código proporcionado por el profesor en clase, para este caso se modificaron los archivos necesarios para ajustarlo a las necesidades del problema propuesto. Para poder ejecutar de forma exitosa el algoritmo es necesario correr el archivo "proyecto_final.m" el cual contiene el programa de ejecución. También fue necesario modificar el archivo "evMOGAresults.m" en donde simplemente se modificó la dimensión de la gráfica de 3 a 2 funciones objetivo. Por último se adjunta también el archivo "mop1.m" el cual contiene las funciones multiobjetivo y es el programa que se utiliza para en el algoritmo ev-MOGA y NNC.

TABLE I. CODIGO UTILIZADO PARA LA SIMULACION CON EL ALGORITMO EV-MOGA

Nombre	**Código utilizado para algoritmo ev-MOGA**
	Código en Matlab

mop1.m	function J=mop1(theta)
	% Constantes definidas
	ta = 24;
	t = 2;
	k = 43;
	% Definiendo las funciones objetivo
	J1=(((exp(-t./(theta(:,1)*theta(:,2))))*theta(:,3))./theta(:,2)) + (((exp(-t./(theta(:,1)*theta(:,2))))*ta)./(theta(:,1)*theta(:,2)));
	J2 = -k.*((((exp(-t./(theta(:,1).*theta(:,2))))).*theta(:,3))./theta(:,2)) + (((exp(-t./(theta(:,1).*theta(:,2))))).*ta)./(theta(:,1).*theta(:,2))));
	J=[J1 J2];

proyecto_final.m	tic clear eMOGA eMOGA.objfun='mop1'; % Nombre de la función m para el cálculo de objetivos eMOGA.objfun_dim=2; % Dimensión del espacio objetivo eMOGA.searchspaceUB=[100 100 100];% Espacio de busqueda alto eMOGA.searchspaceLB=[25 25 25]; % Espacio de busqueda bajo eMOGA.Nind_P= 500; % Individuos por cada P en población eMOGA.Generations= 100; % Numero de generaciones eMOGA.n_div= 200; % Numero de divisiones por cada dimensión

Nombre	**Código utilizado para algoritmo ev-MOGA**
	Código en Matlab
	[pfront,pset,eMOGA]=evMOGA(eMOGA); % Definición del algoritmo ev-MOGA figure(1) plot(pfront(:,1),pfront(:,2),'.r') % plot de J1 y J2 en el frente de pareto title('Grafica del Frente de Pareto ev-MOGA') xlabel('J1'), ylabel('J2') figure(2) plot(pset(:,1),pset(:,2),'.r') % plot de J1 y J2 en el Cnjunto de pareto title('Grafica del Conjunto de Pareto ev-MOGA') xlabel('X'), ylabel('Y') toc

Para el caso de la aplicación del algoritmo NNC en matlab para resolver el problema de optimizacion multiobjetivo se movidificaron los archivos "proyecto_NNC" y CostFuntion "". En el segundo solo se agrego el archivo mop1.m para que fuese tomado en cuenta.

TABLE II. CODIGO UTILIZADO PARA LA SIMULACION CON EL ALGORITMO NNC

Nombre	**Código utilizado para algoritmo NNC**
	Código en Matlab

proyect_NNC.m	close all;
	clc;
	NNCDat.NOBJ = 2; % Numero de objetivos
	NNCDat.NRES = 0; % Numero de restricciones
	NNCDat.NVAR = 3; % Número de variables de decisión
	NNCDat.mop = str2func('CostFunction'); %Archivo con las función de costo
	NNCDat.CostProblem='YourProblem'; %Función de costo en uso.
	NNCDat.FieldD =[[25; 25;25]...
	[100;100;100]]; % Limites de optimización
	NNCDat.mopOPT = str2func('OPTroutine'); % Rutina de optimización en uso,
	%esta llama la rutina de optimización
	%por medio del algoritmo NNC
	NNCDat.InitialGuess=[30 30 30]; % Puntos dentro del intervalo
	% de búsqueda
	NNCDat.AnchorF=[];
	NNCDat.AnchorX=[];
	NNCDat.CounterGEN=0;
	NNCDat.CounterFES=0;
	NNCDat.Card=100; % Numero de divisiones en el plano de utopia
	NNCDat.DominanceFiltering='yes'; % Filtro básico de dominio
	NNCDat.SmartFiltering='yes'; % Filtro inteligente
	NNCDat.RateFilter=[0.1 0.1];
	NNCDat.SaveResults='yes'; % guarda o no los datos mostrados;
	tStart=tic;
	OUT=NNC(NNCDat); %Ejecuta el algoritmo
	tElapsed=toc(tStart);
	figure(3)
	plot(OUT.PSet(:,1),OUT.PSet(:,2),'.r') % Grafica del conjunto de pareto
	title('Grafica del Conjunto de Pareto NNC')
	xlabel('X'), ylabel('Y')

I. RESULTADOS OBTENIDOS

Luego de llevar a cabo la optimización multiobjetivo en Matlab y de recopilar los datos solicitados como el número de iteraciones, tiempo computacional de ejecución, dimensiones del conjunto de Pareto, dimensiones del frente de pareto, relación precio costo y costo computacional por punto; datos mostrados en las tablas III y IV respectivos a los datos de los algoritmos ev-MOGA y NNC. Se procede a graficar el respectivo frente de Pareto y conjunto de Pareto en el espacio de busque asignado.

A. Resultados obtenidos con el algoritmo evMOGA.

TABLE III. TABLA RESUMEN ALGORITMO EV-MOGA

Numero	**Tabla resumen algoritmo evMOGA**	
	Datos del algoritmo	***Resultados***
1)	Numero de iteraciones	100
2)	Tiempo computacional de ejecución	1.295863 s
3)	Dimensiones del conjunto de Pareto	168
4)	Dimensiones del frente de Pareto	168
5)	Relación precio costo	1.68
6)	Costo computacional por punto	0.61

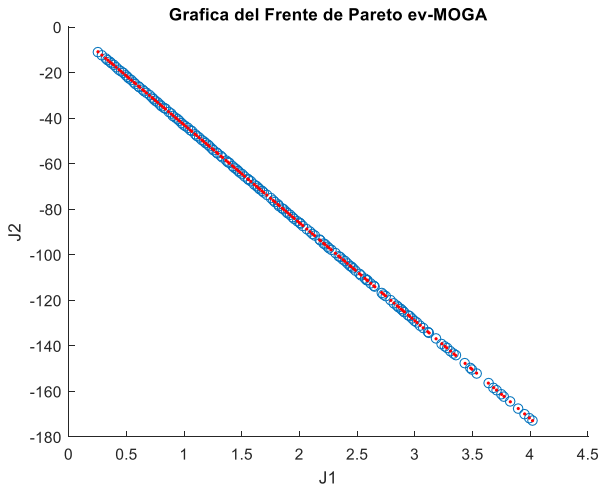

Figura 3: Grafica del frente de Pareto utilizando el algoritmo evMOGA

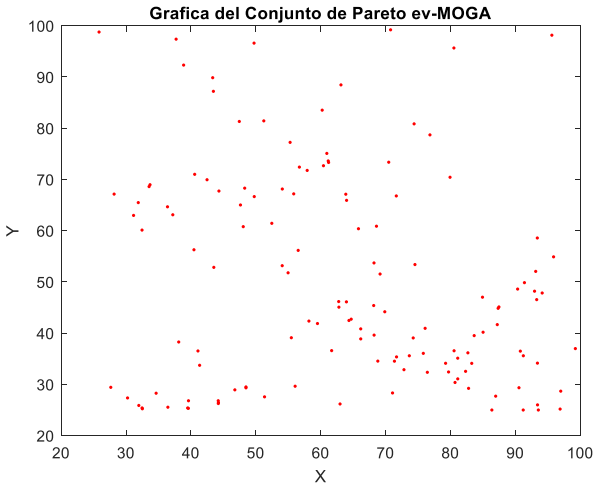

Figura 4: Grafica del conjunto de Pareto utilizando el algoritmo evMOGA.

B. Resultados obtenidos con el algoritmo NNC.

TABLE IV. TABLAN RESUMEN ALGORITMO NNC

Numero	**Tabla resumen algoritmo NNC**	
	Datos del algoritmo	***Resultados***
1)	Numero de iteraciones	2178
2)	Tiempo computacional de ejecución	13.3086 s
3)	Dimensiones del conjunto de Pareto	100
4)	Dimensiones del frente de Pareto	100
5)	Relación precio costo	0.0459
6)	Costo computacional por punto	21.7800

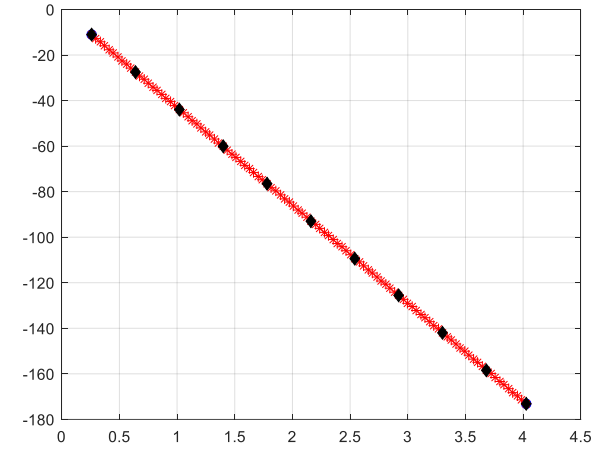

Figura 5: Grafica del frente de Pareto utilizando el algoritmo NNC.

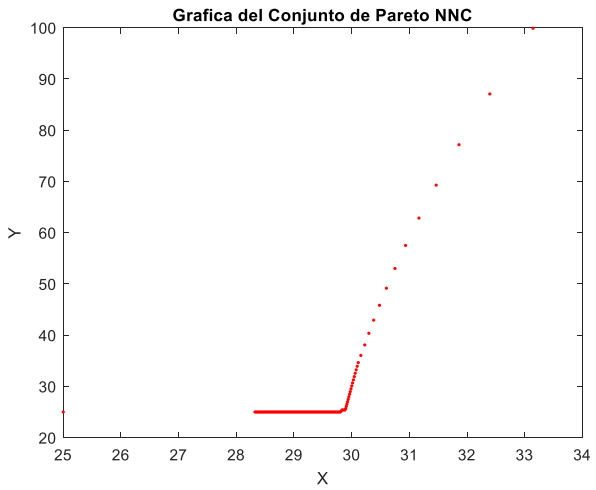

Figura 6: Grafica del frente de Pareto utilizando el algoritmo NNC.

II. ANALISIS Y CONCLUSIONES

Refiriéndonos a la dificultad del planteamiento del problema, podemos ver que a pesar de que el modelo térmico de una maquina eléctrica es, a un nivel general, poco estudiado por los laboratorios de investigación relacionados a la robótica y la automatización, podemos ver que la resolución conlleva a un problema con funciones relativamente sencillas. Sin embargo la mayor cantidad de trabajo la encontramos a la hora de buscar las bases científicas que sustenten el significado del problema y de la búsqueda de soluciones factibles a un problema común usando varias funciones que se opongan entre sí, esto significa buscar funciones que compartan las variables y que al mismo tiempo sean contrarias a la hora de buscar sus resultados. En un principio se planteó resolver la ecuación diferencial resultante del análisis realizado en la figura 1. Para relacionar la temperatura del estator con la temperatura del robot de forma separada, esto llevaría a encontrar variables relacionadas de forma indirecta que conllevarían a funciones opuestas por el principio térmico y la relación del modelo, pero luego de varios análisis y de muchos cálculos matemáticos para poder resolver la ecuación se concluyó que esta ecuación, en realidad, no oponía las funciones de la forma esperada, de modo que se optó por trabajar en un modelo generalizado representado por la figura 2.

Si analizamos la forma del frente de Pareto encontrada por los algoritmos ev-MOGA y NNC (figura 3),(figura 5) podemos ver que ambos pudieron encontrar un frente de Pareto idéntico que se mueve por puntos muy cercanos uno del otro. Sin embargo el algoritmo ev-MOGA fue capaz de encontrar 68 puntos de Pareto adicionales que el algoritmo NNC no pudo encontrar, esto representa una diferencia mostrada en las figura 4 donde se denota una mayor cantidad de punto en el área de muestreo que en la figura 6 que representa los puntos encontrado en el área demuestre del algoritmo NNC.

Tomando en cuenta los datos demostrados en las tablas III y IV podemos ver como el algoritmo NNC requiere un número mucho mayor de iteraciones y un tiempo computacional de ejecución mucho mayor que el utilizado por el algoritmo ev-MOGA para encontrar una menor cantidad de puntos en el espacio de muestra. Además para el caso del algoritmo NNC se muestra un mayor costo computacional por punto, el cual es 21.78 mientras que el utilizado por el algoritmo ev-MOGA es solo de 0.61. La ventaja mostrada del algoritmo ev-MOGA sobre el NNC no solo se reduce a los datos computacionales a la hora de llevar a cabo el cálculo con las funciones objetivo y de trazar las gráficas de los respectivos conjunto de Pareto y frente de Pareto; si no, que a mi parecer, esta ventaja se mantiene también a la hora de implementar el algoritmo en Matlab ya que el código es mucho más sencillo de entender y muestra una mayor facilidad a la hora de variar valores como cantidad de iteraciones o intervalos de búsqueda, también es más sencillo obtener los parámetros de tiempo de computación y los costos que requiere el algoritmo para encontrar el frente y conjunto de Pareto. En el algoritmo NNC es necesario buscar los parámetros que representan estos valores, los cuales son menos evidentes que en ev-MOGA.

El método que obtiene el mejor frente de Pareto es el método que utiliza el algoritmo ev-MOGA, ya que este cumple el objetivo de reducir la temperatura interna del motor para aumentar la velocidad del motor utilizando las mismas funciones objetivos que el algoritmo NNC. Esto se ve reflejado en sus valores objetivos finales de [0.2539381 - 157.6936] para J1 y J2 respectivamente.

A. *Autor*

Recibió su bachillerato en Ingeniería electrónica en 2013 y está cursando la licenciatura en ingeniería electrónica en la Universidad técnica nacional. En 2015 se unió al laboratorio de investigación en Robots Autónomos y Sistemas cognitivos de la Universidad de Costa Rica y ahora es coordinador de la construcción de la base móvil omnidireccional auxiliar y del diseño de una articulación robótica con alta densidad energética para uso futuro con control por impedancia.

REFERENCES

[1] Y. Ito, T. Nakaoka, J. Urata, Y. Nakanishi, K. Okada and M. Inaba, "Design and development of a tendon-driven and axial-driven hybrid humanoid leg with high-power motor driving system," *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, Osaka, 2012, pp. 475-480.

[2] J. Urata, Y. Nakanishi, K. Okada and M. Inaba, "Design of high torque and high speed leg module for high power humanoid," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, 2010, pp. 4497-4502. doi: 10.1109/IROS.2010.5649683

[3] D. j. Shin and B. i. Kwon, "Multi-objective optimal design for in-wheel permanent magnet synchronous motor," *2009 International Conference on Electrical Machines and Systems*, Tokyo, 2009, pp. 1-5. doi: 10.1109/ICEMS.2009.5382882

[4] N. Bianchi, D. Durello and E. Fornasiero, "Multi-objective optimization of a PM Assisted Synchronous Reluctance Machine, including torque and sensorless detection capability," *6th IET International Conference on Power Electronics, Machines and Drives (PEMD 2012)*, Bristol, 2012, pp. 1-6

[5] Y. Ito *et al*., "Development and verification of life-size humanoid with high-output actuation system," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, 2014, pp. 3433-3438. doi: 10.1109/ICRA.2014.6907353

[6] J. Urata, T. Hirose, Y. Namiki, Y. Nakanishi, I. Mizuuchi and M. Inaba, "Thermal control of electrical motors for high-power humanoid robots," *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, 2008, pp. 2047-2052. doi: 10.1109/IROS.2008.4651110.

[7] Ruíz, V. M. A. La red generalizada. *Journal of Tropical Engineering*, *1*(2).

[8] Paine, N., & Sentis, L. (2015, August). Design and Comparative Analysis of a Retrofitted Liquid Cooling System for High-Power Actuators. In *Actuators* (Vol. 4, No. 3, pp. 182-202). Multidisciplinary Digital Publishing Institute.

[9] Messac, A., Ismail-Yahaya, A., & Mattson, C. A. (2003). The normalized normal constraint method for generating the Pareto frontier. *Structural and multidisciplinary optimization*, *25*(2), 86-98.