MATCH
Communications in Mathematical and in Computer Chemistry

MATCH Commun. Math. Comput. Chem. **73** (2015) 175-193

ISSN 0340 - 6253

Computing Lower Bounds for the Kirchhoff Index Via Majorization Techniques*

Gian Paolo Clemente[a], **Alessandra Cornaro**[a]

Department of Mathematics and Econometrics, Catholic University, Milan, Italy

gianpaolo.clemente@unicatt.it, alessandra.cornaro@unicatt.it

(Received July 7, 2014)

Abstract

In this paper, lower bounds for the Kirchhoff index are derived by means of an algorithm developed with $MATLAB^{®}$ software. The procedure localizes the eigenvalues of the transition matrix adapting for the first time a theoretical method, proposed in Bianchi and Torriero (2000, see [4]), based on majorization techniques. Some numerical examples show how sharper bounds can be obtained with respect to those existing in literature.

1 Introduction

The evaluation of the effective resistance between any pair of vertices of a network and the computation of the Kirchhoff index have interest in electric circuit and Probability theory (see [7, 14, 21]). The Kirchhoff index has been also used in Chemistry as an alternative for discriminating among different molecules with similar shapes and structures (see [19]).

In the literature, several techniques (as graph theory, algebra, electric networks and so on) have been explored in order to provide general bounds for this index in terms of invariants of the graph G (see [15, 20, 22]).

In fact, in recent years, there has been an increasing interest in the problem of determining bounds for some relevant topological indicators of graphs. In particular, new

*We thank Monica Bianchi, Anna Torriero and José Luis Palacios for their valuable comments and suggestions.

bounds have been obtained in [2] taking into account additional information on the localization of the eigenvalues of some matrices associated to the graph (i.e Laplacian matrix, Transition matrix, etc.). In this regard, Bianchi et al. [3] have recently proposed a variety of lower and upper bounds for the Kirchhoff index based on majorization techniques, whose major advantage is to provide a unified and flexible approach to recover many bounds in the literature as well as to obtain better ones. For what concerns the localization of real eigenvalues, some well–known theoretical inequalities have been provided in literature (see [18]) which can be used to derive the above mentioned bounds.

Our purpose is instead to compute numerically alternative inequalities involving the localization of the eigenvalues by adapting for the first time a theoretical methodology proposed in Bianchi and Torriero [4] based on nonlinear global optimization problems solved through majorization techniques. By means of these results, we obtain tighter lower bounds for the Kirchhoff Index for some classes of graphs.

In Section 2 some preliminaries are given. In Section 3 we introduce the Kirchhoff index presenting lower bounds proposed in [3] for non-bipartite and bipartite graphs; furthermore, some nonlinear optimization problems, useful for our analysis, are formulated. A computational procedure for determining lower bounds of the Kirchhoff index and some numerical examples are presented in Section 4. In the appendix, we report both the procedures used to generated the graphs and to compute lower bounds.

2 Notation and preliminaries

Let $\mathbf{e^j}$, $j = 1, ...n$, be the fundamental vectors of $\mathbb{R}^n$ and set:

$$\mathbf{s^0} = \mathbf{0}, \quad \mathbf{s^j} = \sum_{i=1}^{j} \mathbf{e^i}, \quad j = 1, ..., n,$$

$$\mathbf{v^n} = \mathbf{0}, \quad \mathbf{v^j} = \sum_{i=j+1}^{n} \mathbf{e^i}, \; j = 0, ..., (n-1).$$

Given two vectors $\mathbf{x}$, $\mathbf{y} \in D = \{\mathbf{x} \in \mathbb{R}^n : x_1 \geq x_2 \geq ... \geq x_n\}$, the majorization order $\mathbf{x} \trianglelefteq \mathbf{y}$ means:

$$\begin{cases} \langle \mathbf{x}, \mathbf{s^k} \rangle \leq \langle \mathbf{y}, \mathbf{s^k} \rangle, \; k = 1, ..., (n-1) \\ \langle \mathbf{x}, \mathbf{s^n} \rangle = \langle \mathbf{y}, \mathbf{s^n} \rangle \end{cases}$$

where $\langle \cdot, \cdot \rangle$ is the inner product in $\mathbb{R}^n$.

Given a subset $S \subseteq \Sigma_a = D \cap \{\mathbf{x} \in \mathbb{R}^n_+ : \langle \mathbf{x}, \mathbf{s^n} \rangle = a\}$, a vector $\mathbf{x}^*(S) \in S$ is said to be the

maximal vector in S with respect to the majorization order if $\mathbf{x} \trianglelefteq \mathbf{x}^*(S)$ for each $\mathbf{x} \in S$. Likewise a vector $\mathbf{x}_*(S) \in S$ is said to be the minimal vector in S with respect to the majorization order if $\mathbf{x}_*(S) \trianglelefteq \mathbf{x}$ for each $\mathbf{x} \in S$.

In the majorization theory, functions which preserve the majorization order play a fundamental role. We recall that a symmetric function ϕ: $A \to \mathbb{R}$, $A \subseteq \mathbb{R}^n$, is said to be Schur-convex on A if $\mathbf{x} \trianglelefteq \mathbf{y}$ implies $\phi(\mathbf{x}) \leq \phi(\mathbf{y})$. If in addition $\phi(\mathbf{x}) < \phi(\mathbf{y})$ for $\mathbf{x} \trianglelefteq \mathbf{y}$ but $\mathbf{x}$ is not a permutation of $\mathbf{y}$, ϕ is said to be strictly Schur-convex on A. A function ϕ is (strictly) Schur-concave on A if $-\phi$ is (strictly) Schur-convex on A.
In particular, given an interval $I \subset \mathbb{R}$, and a (strictly) convex function $g : I \to \mathbb{R}$, the function $\phi(\mathbf{x}) = \sum_{i=1}^{n} g(x_i)$ is (strictly) Schur-convex on $I^n = \underbrace{I \times I \times \cdots \times I}_{n-times}$. The corresponding result holds if g is (strictly) concave on I^n.
In what follows we consider strictly Schur-convex functions of the type $g(\mathbf{x}) = \sum_{i=1}^{n} x_i^p$, where $p > 1$.

Let us now recall some basic graph notations (for more details see [17]).
Let $G = (V, E)$ be a simple, connected, undirected graph where $V = \{1, 2, \ldots, n\}$ is the set of vertices and $E \subseteq V \times V$ the set of edges, $|E| = m$.
The degree sequence of G is denoted by $\pi = (d_1, d_2, .., d_n)$ and it is arranged in non-increasing order $d_1 \geq d_2 \geq \cdots \geq d_n$, where d_i is the degree of vertex i.
Let A be the adjacency matrix of G and $\lambda_1(A) \geq \lambda_2(A) \geq ... \geq \lambda_n(A)$ be the set of its (real) eigenvalues. Given the diagonal matrix D of vertex degrees, the matrix $L = D - A$ is known as the Laplacian matrix of G. Let $\lambda_1(L) \geq \lambda_2(L) \geq ... \geq \lambda_n(L) = 0$ be its eigenvalues.
The transition matrix is $P = D^{-1}A$ and its real eigenvalues are $1 = \lambda_1(P) > \lambda_2(P) \geq \cdots \geq \lambda_n(P) \geq -1$.

3 The Kirchhoff index

The *Kirchhoff index* $K(G)$ of a simple connected graph G was first defined by Klein and Randić in [13] as

$$K(G) = \sum_{i<j} R_{ij},$$

where R_{ij} is the effective resistance between vertices i and j, which can be computed using Ohm's law.

This index has been intensively studied in different fields as Chemistry, Complex Networks, Electric Networks and others.

An alternative expression of the Kirchhoff index is:

$$K(G) = n \sum_{i=1}^{n-1} \frac{1}{\lambda_i(L)}, \tag{1}$$

which involves the non-null eigenvalues of the Laplacian L (see [9], [23]).
If G is d-regular, the transition matrix P is given by $P = I - \frac{1}{d}L$ and

$$\lambda_{n-i+1}(P) = 1 - \frac{\lambda_i(L)}{d}, \quad i = 1, ..., n.$$

In this case, we can rewrite (1) in terms of the eigenvalues of the transition matrix P as follows:

$$K(G) = \frac{n}{d} \sum_{i=2}^{n} \frac{1}{1 - \lambda_i(P)}. \tag{2}$$

For any connected graph G, the following bounds hold (see Corollary 2 in [15]):

$$\left(\frac{n}{d_1}\right) \sum_{i=2}^{n} \left(\frac{1}{1 - \lambda_i(P)}\right) \leq K(G) \leq \left(\frac{n}{d_n}\right) \sum_{i=2}^{n} \left(\frac{1}{1 - \lambda_i(P)}\right). \tag{3}$$

Using a majorization technique that identifies the maximal and minimal vectors of a variety of subsets of $\mathbb{R}^n$ (see [1]), several lower and upper bounds for $K(G)$ have been derived in [3] through the inequalities (3).
In particular, in what follows, we make use of the lower bounds derived in [3], considering additional information on the localization of the eigenvalues of the transition matrix P.

We start exploring the case of non-bipartite graphs. If we have an information of the type

$$\lambda_n(P) \leq -\beta < 0, \tag{4}$$

the bound in terms of β is given by:

$$K(G) \geq \frac{n}{d_1} \left[\frac{1}{1+\beta} + \frac{(n-2)^2}{n-1-\beta}\right]. \tag{5}$$

On the other hand, if we assume

$$\lambda_2(P) \geq \beta > 0, \tag{6}$$

the bound is given by:

$$K(G) \geq \frac{n}{d_1} \left[\frac{1}{1-\beta} + \frac{(n-2)^2}{n-1+\beta}\right]. \tag{7}$$

In case of bipartite graphs, since $\lambda_n(P) = -1$,, the relation (4) is not significant. Hence if we set $\beta = 1$ in (5), we derive the same bound given in [16], Corollary 3 (for further details see Section 4.2 later on). Relation (6) is instead equivalent to $\lambda_{n-1}(P) \leq -\beta < 0$, by the symmetry of the spectrum. Thus bound (7) can be rewritten as:

$$K(G) \geq \frac{n}{d_1}\left[\frac{\beta-3}{2(\beta-1)} + \frac{(n-3)^2}{n-3+\beta}\right]. \tag{8}$$

In order to obtain the value of β, we make use of nonlinear global optimization problems solved through majorization techniques studied in [4]. The following Theorems 1 and 2 allow us to compute lower and upper bounds for the eigenvalues of the transition matrix P and to derive, in particular, the lower bounds (5) and (7) for the Kirchhoff Index.

Let

$$\nu_i = 1 - \lambda_{n-i+1}(P),\ i = 1, \cdots, (n-1).$$

For the vector $\boldsymbol{\nu} \in \mathbb{R}^{n-1}$ we have

$$0 < \nu_{n-1} \leq \nu_{n-2} \leq \cdots \leq \nu_1 \leq 2$$

and $\sum_{i=1}^{n-1} \nu_i = n$ since

$$\mathrm{tr}(P) = \sum_{i=1}^{n} \lambda_i(P) = 0 \Rightarrow \sum_{i=2}^{n} \lambda_i(P) = -1.$$

We now face the set

$$S_b = \{\boldsymbol{\nu} \in \mathbb{R}_+^{n-1} : \sum_{i=1}^{n-1} \nu_i = n, g(\boldsymbol{\nu}) = \sum_{i=1}^{n-1} \nu_i^p = b\}.$$

where p is an integer greater than 1.

The following fundamental lemma holds (see Lemma 5.1 in [5]):

Lemma 1. *Fix $b \in \mathbb{R}$ and consider the set S_b. Then either $b = \frac{n^p}{(n-1)^{p-1}}$ or there exists a unique integer $1 \leq h^* < (n-1)$ such that:*

$$\frac{n^p}{(h^*+1)^{p-1}} < b \leq \frac{n^p}{(h^*)^{p-1}}, \tag{9}$$

where $h^* = \left\lfloor \sqrt[p-1]{\frac{n^p}{b}} \right\rfloor$.

Proof. Let $\boldsymbol{w^h} = \frac{1}{h}\boldsymbol{s^h}$, $h = 1, ..., (n-1)$. We have $\boldsymbol{w^h} \in \Sigma_1 = \{\boldsymbol{\nu} \in \mathbb{R}_+^{n-1}, \langle \boldsymbol{\nu}, \boldsymbol{s^{n-1}} \rangle = 1\}$ and $\boldsymbol{w^{h+1}} \trianglelefteq \boldsymbol{w^h}$, $h = 1, ..., (n-2)$.
Furthermore, $\boldsymbol{w^1} = \boldsymbol{e^1}$ and $\boldsymbol{w^{n-1}} = \underbrace{\left(\frac{1}{n-1}, ..., \frac{1}{n-1}\right)}_{n-1}$ are the maximal and the minimal elements of the set Σ_1 with respect to the majorization order. Since $g(\boldsymbol{\nu}) = \sum_{i=1}^{n-1} \nu_i^p$ is strictly Schur-convex, we obtain:

$$\left(\frac{1}{n-1}\right)^{p-1} < \left(\frac{1}{n-2}\right)^{p-1} < ... < 1.$$

Since $S_b \neq \emptyset$, there exists $\hat{\boldsymbol{\nu}} \in S_b$. We consider $\hat{\boldsymbol{w}} = \left(\frac{1}{n}\right)\hat{\boldsymbol{\nu}} \in \Sigma_1$. From $g(\hat{\boldsymbol{\nu}}) = b$ and by homogeneity we have

$$g(\hat{\boldsymbol{w}}) = \left(\frac{1}{n}\right)^p b = \frac{b}{n^p}. \tag{10}$$

1. If $\hat{\boldsymbol{w}} = \boldsymbol{w^{n-1}}$, then $g(\hat{\boldsymbol{w}}) = \left(\frac{1}{n-1}\right)^{p-1}$ and from equation (10) we have $\left(\frac{1}{n-1}\right)^{p-1} = \frac{b}{n^p}$. Hence, $\frac{n^p}{(n-1)^{p-1}} = b$.

2. If $\hat{\boldsymbol{w}} \neq \boldsymbol{w^{n-1}}$ and $\hat{\boldsymbol{w}} \neq \boldsymbol{w^1}$, from the condition $\boldsymbol{w^{n-1}} \trianglelefteq \hat{\boldsymbol{w}} \trianglelefteq \boldsymbol{w^1}$ we deduce $\frac{1}{(n-1)^{p-1}} < \frac{b}{n^p} \leq 1$. Since $g(\hat{\boldsymbol{w}}) = \frac{b}{n^p}$, there exist a unique integer h^*, where $1 < h^* < (n-1)$, such that $\frac{1}{(h^*+1)^{p-1}} < \frac{b}{n^p} \leq \frac{1}{(h^*)^{p-1}}$ and we get the condition (9).

3. If $\hat{\boldsymbol{w}} = \boldsymbol{w^1} = \boldsymbol{e^1}$ we have that $g(\hat{\boldsymbol{w}}) = 1$ and from (10), $b = n^p$ so that condition (9) is satisfied for $h^* = 1$.

To complete the proof notice that from (9), $\frac{1}{(h^*+1)^{p-1}} < \frac{b}{n^p} \leq \frac{1}{(h^*)^{p-1}}$, $h^* \leq \sqrt[p-1]{\frac{n^p}{b}} < h^*+1$ and thus $h^* = \left\lfloor \sqrt[p-1]{\frac{n^p}{b}} \right\rfloor$.

□

We can now deduce upper and lower bounds for ν_h by solving the following optimization problems:

$$\max\ (\nu_h) \ \text{ subject to } \boldsymbol{\nu} \in S_b \qquad P(h)$$

$$\min\ (\nu_h) \ \text{ subject to } \boldsymbol{\nu} \in S_b \qquad P^*(h)$$

Theorem 1. *The solution of the optimization problem $P(h)$ is $(\frac{n}{n-1})$ if $b = \frac{n^p}{(n-1)^{p-1}}$. If $b \neq \frac{n^p}{(n-1)^{p-1}}$, the solution of the optimization problem $P(h)$ is α^* where*

1. *for* $h > h^*$, α^* *is the unique root of the equation*

$$f(\alpha, p) = (h-1)\alpha^p + (n - h\alpha + \alpha)^p - b = 0 \tag{11}$$

in $I = \left(0, \frac{n}{h}\right]$;

2. *for* $h \leq h^*$, α^* *is the unique root of the equation*

$$f(\alpha, p) = h\alpha^p + (n-1-h)\frac{(n-h\alpha)^p}{(n-1-h)^p} - b = 0 \tag{12}$$

in $I = \left(\frac{n}{n-1}, \frac{n}{h}\right]$.

Theorem 2. *The solution of the optimization problem* $P^*(h)$ *is* $(\frac{n}{n-1})$ *if* $b = \frac{n^p}{(n-1)^{p-1}}$.

If $b \neq \frac{n^p}{(n-1)^{p-1}}$, *the solution of the optimization problem* $P^*(h)$ *is* α^* *where*

1. *for* $h = 1$, α^* *is the unique root of the equation*

$$f(\alpha, p) = h^*\alpha^p + (n - h^*\alpha)^p - b = 0 \tag{13}$$

in $I = \left(\frac{n}{h^*+1}, \frac{n}{h^*}\right]$;

2. *for* $1 < h \leq (h^*+1)$, α^* *is the unique root of the equation*

$$f(\alpha, p) = (n-h)\alpha^p + (h-1)\frac{(n-(n-h)\alpha)^p}{(h-1)^p} - b = 0 \tag{14}$$

in $I = (0, \frac{n}{n-1}]$;

3. *for* $h > (h^*+1)$, α^* *is zero.*

Theorems 1 and 2 allow us to numerically compute, respectively, the upper and lower bounds α^* of ν_h.

It is noteworthy to underline that the suitable change of variable stating the relation between $\lambda_{n-h+1}(P)$ and ν_h leads to derivations of β in (4) and (6). Indeed from $\nu_1 \geq (1+\beta) > 1$ follows $\lambda_n(P) \leq -\beta$, while from $\nu_{n-1} \leq (1-\beta) < 1$ follows $\lambda_2(P) \geq \beta$. By using these bounds we obtain lower bounds of the Kirchhoff Index via formulae (5) and (7).

4 Numerical examples

In order to compute the lower bound of the Kirchhoff Index, a computational procedure through $MATLAB^{®}$ software has been implemented.

To this aim, for every $p \geq 2$, we can rewrite the function $g(\boldsymbol{\nu}) = \sum_{i=1}^{n-1} \nu_i^p$ as follows:

$$\begin{aligned} g(\boldsymbol{\nu}) &= \sum_{i=1}^{n-1} \nu_i^p = \sum_{i=1}^{n-1} \left(\sum_{k=0}^{p} \binom{p}{k} (1)^k \left(-\lambda_{n-i+1}(P)\right)^{p-k} \right) = \\ &= \sum_{k=0}^{p} \binom{p}{k} (-1)^{p-k} \sum_{i=1}^{n-1} \left(\lambda_{n-i+1}(P)\right)^{p-k} = \\ &= \sum_{k=0}^{p} \binom{p}{k} (-1)^{p-k} \left[\sum_{i=1}^{n} \left(\lambda_i(P)\right)^{p-k} - \left(\lambda_1(P)\right)^{p-k} \right] = \\ &= \sum_{k=0}^{p} \binom{p}{k} (-1)^{p-k} \left[tr\left(P^{p-k}\right) - 1 \right] = \\ &= \sum_{k=0}^{p} \binom{p}{k} (-1)^{p-k} \, tr\left(P^{p-k}\right). \end{aligned} \tag{15}$$

Now we describe the procedure used to compute the upper and lower bounds of ν_h for $1 \leq h \leq (n-1)$. For our aim, we consider only the upper bound for ν_{n-1} and the lower bound of ν_1.

It is noteworthy that the following procedure is developed for simple, connected and non-bipartite graphs.

Step 0 generate[1] randomly a simple connected graph G by fixing the number of vertices;

I) Upper Bound for ν_h (see Theorem 1)

Step 1 repeat the following steps for several values of p (with $p \in \mathbb{N}$ and $p > 1$) until a fixed value p^* is reached and for each h (with $1 \leq h \leq (n-1)$);

a) evaluate a equal to the number n of vertices of G and b equal to the right hand side of (15);

b) if $b = \frac{a^p}{(n-1)^{p-1}}$, then the bound is equal to $(\frac{a}{n-1})$ and the procedure stops; otherwise compute $h^* = \left\lfloor \sqrt[p-1]{\frac{a^p}{b}} \right\rfloor$;

c) compare h^* to h in order to choose the proper equation (11) or (12);

[1]The graph is generated by using a procedure in $MATLAB^{®}$ that allows to derive a simple and connected graph. For more details see the Appendix.

d) evaluate the unique root $\alpha^*_{h,p}$ of equation (11) or (12);

Step 2 choose the minimum upper bound α^*_h of ν_h among all the $\alpha^*_{h,p}$ $(2 \leq p \leq p^*)$;

Step 3 pick the value of the upper bound α^*_{n-1} of ν_{n-1} in order to consider the particular case $h = n-1$;

Step 4 set β equal to $1-\alpha^*_{n-1}$ in order to evaluate the lower bound (7) and compare to other bounds existing in literature.

II) Lower Bound for ν_h (see Theorem 2)

Step 1 repeat the following steps for several values of p (with $p \in \mathbb{N}$ and $p > 1$) until a fixed value p^* is reached and for each h (with $1 \leq h \leq (n-1)$);

a) evaluate a equal to the number n of vertices of G and b equal to the right hand side of (15);

b) if $b = \frac{a^p}{(n-1)^{p-1}}$, then the bound is equal to $(\frac{a}{n-1})$ and the procedure stops, otherwise compute $h^* = \left\lfloor \sqrt[p-1]{\frac{a^p}{b}} \right\rfloor$;

c1) for $h = 1$ evaluate the unique root $\alpha^*_{1,p}$ of the equation (13);

c2) for $1 < h \leq (h^*+1)$ evaluate instead the unique root $\alpha^*_{h,p}$ of the equation (14);

c3) for $h > (h^*+1)$ set $\alpha^*_{h,p}$ equal to zero;

Step 2 choose the maximum lower bound α^*_h of ν_h among all the $\alpha^*_{h,p}$ $(2 \leq p \leq p^*$, $1 \leq h \leq (h^*+1))$;

Step 3 pick the value of the lower bound α^*_1 of ν_1 in order to consider the particular case $h = 1$;

Step 4 set β equal to $\alpha^*_1 - 1$ in order to evaluate the lower bound (5) and compare to other bounds existing in literature.

For bipartite graphs, only case (I) is significant, and in the step (4) β is used in order to evaluate the proper bound (8).

4.1 Non-bipartite graphs

We now focus only on non-bipartite graphs.

Table 1 shows the comparison between the bounds (5) and (7) with the bounds

$$K(G) \geq \frac{(n-1)^2}{d_1} \tag{16}$$

in [15] and

$$K(G) \geq \frac{n}{d_1}\left[\frac{1}{1+\frac{\sigma}{\sqrt{n-1}}} + \frac{(n-2)^2}{n-1-\frac{\sigma}{\sqrt{n-1}}}\right] \tag{17}$$

in [3] where $\sigma = \sqrt{\frac{tr(P^2)}{n} - \left(\frac{tr(P)}{n}\right)^2} = \sqrt{\frac{tr(P^2)}{n}}$.
These formulas represent tight lower bounds for the Kirchhoff Index of n-vertex graph in terms of n and its maximal degree d_1. For further details see the corresponding references.

The bounds have been computed for different number of vertices (from 4 to 1000) and assuming $p^* = 150$. The choice of p^* should be based on a best compromise between improvement of the results and computational time. Higher values of p need longer cpu time but not always provide sharper bounds.

Notice (see Table 1) that both lower bounds (5) and (7) are tighter than bounds (16) and (17), but as n increases, slighter and slighter differences are observed. For a better readability, in the lower part of the table we report the absolute value of the difference between the bound and the Kirchhoff Index and the relative errors, obtained dividing the absolute value by $K(G)$. Finally, for a better comparison, we provide the ratio between the errors of bound (17) and bound (5).

n	d_1	m	$K(G)$	bound (5)	bound (7)	bound (16)	bound (17)
4	2	3	10.0	5.51	4.82	4.50	4.52
5	3	6	10.2	5.84	5.42	5.33	5.34
10	6	23	21.5	14.34	13.52	13.50	13.50
20	15	105	37.4	25.12	24.07	24.07	24.07
30	21	223	61.3	41.12	40.05	40.05	40.05
50	30	612	101.5	81.19	80.03	80.03	80.03
100	65	2553	195.8	151.33	150.78	150.78	150.78
200	116	9907	403.7	341.70	341.39	341.39	341.39
300	173	22257	606.4	516.95	516.77	516.77	516.77
500	285	62571	998.8	873.71	873.69	873.69	873.69
1000	548	249224	2,006.2	1,821.18	1,821.17	1,821.17	1,821.17

	absolute errors (e)		relative errors (r)		ratio of abs. errors
n	e(5)	e(17)	r(5)	r(17)	e(17)/e(5)
4	4.49	5.18	44.95%	51.84%	1.15
5	4.41	4.83	43.05%	47.15%	1.10
10	7.16	7.98	33.29%	37.10%	1.11
20	12.25	13.30	32.79%	35.60%	1.09
30	20.21	21.28	32.95%	34.69%	1.05
50	20.36	21.51	20.05%	21.19%	1.06
100	44.51	45.05	22.73%	23.01%	1.01
200	62.04	62.35	15.37%	15.44%	1.00
300	89.47	89.66	14.75%	14.78%	1.00
500	125.08	125.09	12.52%	12.52%	1.00
1000	185.04	185.05	9.22%	9.22 %	1.00

Table 1: Lower bounds for $K(G)$ and errors

In this regard, Figure 1 shows that both the upper bound of ν_{n-1}, used in (7), and the value of $1 + \frac{\sigma}{\sqrt{n-1}}$ in (17) tend to 1 providing similar lower bounds for $K(G)$. Conversely

higher values of ν_1 lead to an improvement of the results, as bound (5) shows. However, when the number of vertices is really high (roughly greater than 500), also the lower bound of ν_1 assumes values almost equal to one leading to bounds really similar to equations (7) and (17).

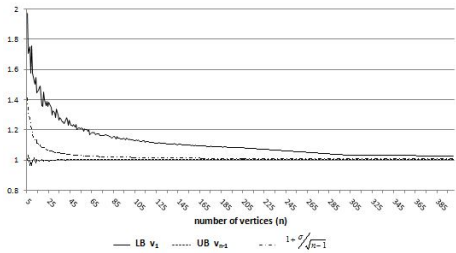

Figure 1: Lower bound of ν_1, Upper bound of ν_{n-1} and $1 + \frac{\sigma}{\sqrt{n-1}}$ used in (17)

It should be noticed that the results of the procedure used to generate the graph are based on a probability of existence of edges q equal to 0.5 following the Erdös-Rényi model $G_{ER}(n, q)$ (see [6, 8, 10, 11]). In this model, the graph is constructed by connecting nodes randomly such that edges are included with probability independent from every other edge. We obtain that the generated graphs have a number of edges not far from the half of its maximum value as proved in the literature (see for example [12]).

Furthermore, the parameter q can be thought of as a weighting function. As q increases from 0 to 1, the model becomes more and more likely to include graphs with more edges and less and less likely to include graphs with fewer edges. In this regard, we assign several values of q moving from the default value of 0.5. Table 2 shows that bound (5) performs better than other bounds also when lower values of probability q are considered. For sake of simplicity we report only the values of bound (17) for comparison, while other bounds provide worse values in this case too. Moreover, it is confirmed that the density of the graphs increases as long as greater probabilities are considered. As expected, for dense graphs, we observe a lower value for the Kirchhoff Index. Finally, we can observe that the performance of both bounds worsens as q get smaller as relative errors show. However it could be noticed how for large graphs we have a greater improvement for lower values of q.

	q = 0.1						**q = 0.3**					
n	d_1	m	$K(G)$	r(5)	r(17)	e(17)/e(5)	d_1	m	$K(G)$	r(5)	r(17)	e(17)/e(5)
4	2	3	10.00	44.95%	54.82%	1.22	2	3	10.00	47.68%	51.84%	1.09
5	3	4	18.00	64.56%	70.29%	1.09	3	6	10.25	43.05%	47.15%	1.10
10	3	10	96.17	68.06%	71.88%	1.06	5	14	51.91	64.95%	68.67%	1.06
20	5	27	293.72	73.23%	75.41%	1.03	9	55	89.06	51.30%	54.94%	1.07
30	9	55	454.84	78.21%	79.45%	1.02	13	135	108.23	36.96%	40.22%	1.09
50	10	129	754.25	67.04%	68.17%	1.02	25	377	176.42	44.74%	45.56%	1.02
100	19	515	1,132.79	53.49%	54.46%	1.02	40	1503	342.19	27.98%	28.40%	1.01
200	37	2017	2,168.41	49.88%	50.64%	1.02	79	5956	681.75	26.37%	26.47%	1.00
300	42	4458	3,213.32	33.08%	33.76%	1.02	111	13368	1,019.16	20.93%	20.97%	1.00
500	70	12361	5,228.61	31.62%	31.97%	1.01	184	37435	1,678.17	19.36%	19.36%	1.00
1000	134	49887	10,181.79	26.67%	26.85%	1.01	343	149718	3,348.29	13.10%	13.10%	1.00

	q = 0.7						**q = 0.9**					
n	d_1	m	$K(G)$	r(5)	r(17)	e(17)/e(5)	d_1	m	$K(G)$	r(5)	r(17)	e(17)/e(5)
4	2	3	10.00	44.95%	51.84%	1.15	3	5	4.00	20.44%	24.15%	1.18
5	3	6	10.25	40.31%	47.15%	1.17	4	9	4.67	13.11%	14.08%	1.07
10	8	33	13.18	20.52%	23.12%	1.13	9	39	10.92	17.13%	17.60%	1.03
20	17	131	29.05	24.99%	26.90%	1.08	19	172	21.20	10.03%	10.37%	1.03
30	23	289	45.43	18.15%	19.51%	1.07	29	397	31.96	9.05%	9.26%	1.02
50	41	831	73.61	19.73%	20.45%	1.04	48	1105	54.57	8.22%	8.34%	1.01
100	84	3515	140.63	16.74%	17.03%	1.02	96	4454	110.27	7.37%	7.42%	1.01
200	156	13944	285.18	10.91%	10.98%	1.01	189	17930	221.09	5.22%	5.23%	1.00
300	234	31281	429.93	11.10%	11.14%	1.00	281	40345	332.63	4.35%	4.35%	1.00
500	384	87410	713.32	9.09%	9.10%	1.00	469	112346	554.34	4.22%	4.23%	1.00
1000	744	349583	1,428.70	6.11%	6.11%	1.00	926	449725	1,109.82	2.89%	2.89%	1.00

Table 2: Relative errors for Lower bounds for $K(G)$ of randomly generated graphs with several probabilities of existence of edges

We observe then that the proposed procedure improves existing bounds according to randomly generated graphs, but the improvement appears reduced for very large graphs. Furthermore the complexity of the algorithm increases as either the number of vertices or the value of p^* increases. In this regard, it could be noteworthy that both the upper bound of ν_{n-1} and the lower bound of ν_1 have been obtained by picking, respectively, the minimum and the maximum value observed among different p, allowing us to get the tighter one. Numerical analysis shows that usually the upper bounds of ν_h with $h = 1, ..., (n-1)$ improve monotonically, when p increases, down to a minimum value. After this threshold, further values of p are not significant for the performed procedure. In particular, as long as ν_{n-1} is considered, this minimum value is reached for lower values of p (usually equal to 2), while to achieve the best bound for the first values of ν_h greater p are needed.

For what concerns the lower bound of ν_1, we observe a monotonic behaviour when p increases. However, for very high values of p (for example for $p > 100$), we obtain negligible improvements.

In Table 3, we report main results derived by generating $1,000$ sampled graphs for some combinations of n and q. A sufficiently large value of p^* has been picked in order to assure an improvement not lower than 10^{-3} for the lower bound of ν_1. We provide the average and the standard deviation of the cpu time[2] needed for graph generation

[2]Results have been obtained with a Desktop PC (Intel Core 2 Duo Processor E7500, 2.93 GHz and 4GB RAM) and the time has been derived by evaluating in a separate way the lower bound of ν_1 and

and bounds evaluation (derived for $2 \leq p \leq p^*$). Furthermore, the minimum and the maximum time value among the samples are included.

Last three columns in the table display the average value, the minimum and the maximum of the ratios between the errors of bound (17) and bound (5).

n	q	p^*	average time	standard dev. (time)	min(time)	max(time)	average ratio	min ratio	max ratio
10	0.1	50	0.18	0.05	0.14	0.50	103.84%	101.13%	111.86%
10	0.3	50	0.15	0.01	0.14	0.18	107.25%	101.99%	120.64%
10	0.5	50	0.15	0.01	0.14	0.20	108.86%	101.77%	125.82%
10	0.7	50	0.14	0.01	0.13	0.18	110.94%	101.92%	127.80%
10	0.9	50	0.15	0.01	0.13	0.19	105.54%	100.00%	118.20%
20	0.1	50	0.38	0.14	0.25	1.06	102.19%	100.00%	104.18%
20	0.3	50	0.30	0.09	0.22	0.64	107.57%	100.00%	119.69%
20	0.5	50	0.27	0.10	0.22	1.07	105.39%	100.00%	111.08%
20	0.7	50	0.22	0.07	0.17	0.50	105.82%	100.00%	111.90%
20	0.9	50	0.20	0.05	0.16	0.52	102.40%	100.00%	104.94%
50	0.1	50	0.56	0.58	0.40	5.55	101.66%	100.00%	103.31%
50	0.3	50	0.55	0.28	0.41	2.20	102.39%	100.00%	103.91%
50	0.5	50	0.48	0.19	0.40	1.70	103.99%	100.00%	106.01%
50	0.7	50	0.52	0.17	0.40	1.36	104.28%	100.00%	106.79%
50	0.9	50	0.54	0.30	0.41	3.03	101.40%	100.00%	102.21%
100	0.1	50	1.98	0.56	1.63	4.90	101.77%	100.00%	102.59%
100	0.3	50	1.98	0.48	1.63	4.08	101.33%	100.00%	102.07%
100	0.5	50	1.97	0.58	1.51	5.16	102.10%	100.00%	103.13%
100	0.7	50	1.86	0.47	1.48	3.57	102.16%	100.00%	103.08%
100	0.9	50	1.91	0.49	1.51	3.88	100.67%	100.00%	101.09%
200	0.1	40	9.25	2.30	6.19	16.21	102.08%	100.00%	102.71%
200	0.1	20	2.03	0.82	1.29	5.61	101.99%	100.00%	102.73%
200	0.3	40	11.68	2.93	6.81	20.81	100.43%	100.00%	100.58%
200	0.3	20	3.03	0.85	1.29	7.61	100.33%	100.00%	100.57%
200	0.5	40	13.51	1.24	10.75	16.19	100.69%	100.48%	100.83%
200	0.5	20	2.17	0.63	1.52	3.77	100.58%	100.37%	100.76%
200	0.7	40	10.83	1.26	8.61	14.38	100.68%	100.57%	100.86%
200	0.7	20	2.16	0.59	1.54	3.57	100.60%	100.39%	100.93%
200	0.9	40	10.94	0.74	9.50	12.06	100.22%	100.14%	100.27%
200	0.9	20	2.27	0.63	1.58	3.54	100.11%	100.04%	100.17%
1000	0.1	20	151.57	8.14	146.20	173.80	100.31%	100.27%	100.35%
1000	0.1	2	2.22	0.59	1.80	4.57	100.14%	100.00%	100.18%
1000	0.3	20	151.86	9.52	142.44	171.14	100.03%	100.03%	100.04%
1000	0.3	2	1.90	0.10	1.83	2.36	100.00%	100.00%	100.00%
1000	0.5	20	138.44	0.77	136.88	139.37	100.05%	100.04%	100.07%
1000	0.5	2	2.16	0.25	2.04	3.68	100.01%	100.00%	100.01%
1000	0.7	20	161.66	7.32	151.96	170.51	100.01%	100.00%	100.01%
1000	0.7	2	2.45	0.45	2.05	4.06	100.01%	100.00%	100.01%
1000	0.9	20	149.46	5.14	142.08	157.01	100.01%	100.00%	100.01%
1000	0.9	2	2.25	0.32	2.02	3.45	100.01%	100.00%	100.01%

Table 3: Cpu time and bounds improvements for several $G(n, q)$ generated graphs

It could be noticed how the bounds are calculated in less than one second when the number of vertices is lower than 100 for all the generated graphs. In these cases, the average ratio is always greater than 1 and bound (17) never performs better than bound (5). Time is obviously increasing for larger graphs, but the procedure is longer than one minute only when the number of vertices is roughly greater than 350-400 and p^* is at least equal to 30-40 (for the sake of simplicity only main results are reported in Table 3). According to large graphs, very low values of p (for example $p = 2$ in Table 3), leads to very low cpu time (roughly 2 seconds for 1,000 vertices) and to smaller improvements. However for these graphs, the procedure could be revised for bounds (5) fixing as a prior a sufficiently large value of p in order to avoid the iterative process and to reduce the computational time. For example, for the generated graph with 1000 vertices, if we fix

the upper bound of ν_{n-1}.

directly $p = p^* = 20$, we derive the lower bound (5) in approximately 30 seconds instead of 150 seconds obtained by the iterative procedure (see Table 3).

Table 4 shows relative errors and ratios for bounds computed by using $G(n, m)$ Erdös–Rényi variant where a random graph is picked out of those with n nodes and m edges. A significant sample $(1,000)$ of random graphs is here derived and mean, standard deviation and variability coefficient (CV) of relative errors are detailed. An interesting scenario is depicted since by increasing the number of edges, we observe lower values of sample mean and sample variance. Also for $G(n, m)$ variant, bound (5) performs better but ratios tend to 1 for greater n.

n	m	average $r(5)$	standard dev. $r(5)$	CV($r(5)$)	average ratio	min ratio	max ratio
10	15	0.61	0.08	0.13	105.93%	102.69%	112.07%
10	20	0.50	0.09	0.18	105.33%	102.17%	112.33%
10	40	0.13	0.01	0.05	102.93%	102.47%	103.23%
50	100	0.74	0.04	0.05	100.97%	100.59%	101.61%
50	500	0.32	0.04	0.12	103.05%	101.96%	104.60%
50	1000	0.13	0.02	0.16	100.89%	100.54%	101.18%
100	200	0.78	0.03	0.03	100.61%	100.41%	100.81%
100	1000	0.39	0.04	0.09	101.64%	100.92%	102.29%
100	2000	0.25	0.03	0.12	101.70%	101.02%	102.42%
100	4000	0.11	0.02	0.14	100.42%	100.26%	100.55%
500	1500	0.74	0.03	0.04	100.01%	100.01%	100.01%
500	5000	0.48	0.03	0.06	100.01%	100.01%	100.02%
500	10000	0.36	0.03	0.09	100.01%	100.01%	100.02%
500	50000	0.15	0.02	0.10	100.02%	100.01%	100.02%
500	100000	0.06	0.01	0.11	100.03%	100.02%	100.03%
1000	5000	0.63	0.02	0.04	100.00%	100.00%	100.01%
1000	10000	0.50	0.03	0.06	100.00%	100.00%	100.00%
1000	100000	0.18	0.01	0.08	100.00%	100.00%	100.01%

Table 4: Relative errors and ratios of bounds according to $G(n, m)$ generated graphs

4.2 Bipartite graphs

We bring now our attention to bipartite graphs.

To this aim we compare bound (8) with bound (16) and bound

$$K(G) \geq \frac{n(2n-3)}{2d_1} \tag{18}$$

in [16], Corollary 3.

Previous bounds have been derived both for path graphs, P_n, and trees, T, as reported in Table 5. The procedure is not significant for the star graph. In fact, being a particular case of complete bipartite graph $K_{1,n-1}$, $\lambda_2(P)$ is equal to zero and then by formula (8) we get the bound (18).

Non-bipartite graphs display a similar pattern to bipartite graphs since bound (8) provides an improvement respect to both bounds (16) and (18). Table 5 depicts slighter differences for larger graphs in this case too. However it could be noticed how the absolute improvement of bound (8) respect to other bounds is greater than non-bipartite graphs

for high number of vertices. On the other hand, all the bounds are far from the exact value of $K(G)$ with increasing relative errors.

Path Graphs $\mathbf{P_n}$									
n	d_1	m	$K(G)$	Bound (8)	Bound(16)	Bound(18)	r(8)	r(18)	e(18)/e(8)
4	2	3	10	5.24	4.50	5.00	47.61%	50.00%	104.77%
5	2	4	20	9.05	8.00	8.75	54.76%	56.25%	103.41%
10	2	9	165	43.01	40.50	42.50	73.93%	74.24%	101.20%
20	2	19	1,330	185.81	180.50	185.00	86.03%	86.09%	100.44%
30	2	29	4,495	428.56	420.50	427.50	90.47%	90.49%	100.25%
50	2	49	20,825	1,214.00	1,200.50	1,212.50	94.17%	94.18%	100.12%
100	2	99	166,650	4,927.46	4,900.50	4,925.00	97.04%	97.04%	100.05%
200	2	199	1.33E+06	19,854.16	19,800.50	19,850.00	98.51%	98.51%	100.02%
300	2	299	4.50E+06	44,780.76	44,700.50	44,775.00	99.00%	99.00%	100.01%
500	2	499	2.08E+07	124,633.80	124,500.50	124,625.00	99.40%	99.40%	100.01%
1000	2	999	1.67E+08	499,266.01	499,000.50	499,250.00	99.70%	99.70%	100.00%
Trees T									
n	d_1	m	$K(G)$	Bound(8)	Bound(16)	Bound(18)	r(8)	r(18)	e(18)/e(8)
4	2	3	10	5.24	4.50	5.00	47.61%	50.00%	104.77%
5	2	4	20	9.05	8.00	8.75	54.76%	56.25%	103.41%
10	3	9	150	28.66	27.00	28.33	80.89%	81.11%	101.16%
20	4	19	783	92.88	90.25	92.50	88.14%	88.19%	100.41%
30	5	29	1,890	171.40	168.20	171.00	90.93%	90.95%	100.24%
50	7	49	5,885	346.83	343.00	346.43	94.11%	94.11%	100.12%
100	7	99	33,798	1,407.82	1,400.14	1,407.14	95.83%	95.84%	100.05%
200	7	199	1.63E+05	5,672.59	5,657.29	5,671.43	96.51%	96.51%	100.02%
300	8	299	3.62E+05	11,195.16	11,175.13	11,193.75	96.91%	96.91%	100.01%
500	12	499	1.14E+06	20,772.28	20,750.08	20,770.83	98.18%	98.18%	100.01%
1000	12	999	5.36E+06	83,210.97	83,166.75	83,208.33	98.45%	98.45%	100.00%

Table 5: Lower Bounds and errors for $K(G)$ for path graphs and trees

Finally, Table 6 reports mean and standard deviation of cpu time and ratios according to 1,000 samples of trees for each combination. In this case, we have a lower cpu time for the generation of the graphs leading to observe a lower overall cpu time than non-bipartite graphs.

n	p^*	average time	standard dev. (time)	min(time)	max(time)	average ratio	min ratio	max ratio
10	50	0.19	0.01	0.18	0.26	101.12%	100.97%	101.20%
20	50	0.38	0.23	0.21	1.35	100.48%	100.42%	100.53%
50	50	0.60	0.31	0.39	2.86	100.17%	100.12%	100.22%
100	50	2.35	1.32	1.44	12.30	100.14%	100.00%	100.53%
200	40	6.83	2.17	5.56	18.41	100.07%	100.00%	100.12%
200	20	1.99	0.72	1.36	5.19	100.02%	100.00%	100.07%
1000	20	142.94	14.38	133.73	201.35	100.01%	100.00%	100.05%
1000	2	2.26	0.39	1.73	3.71	100.01%	100.00%	100.05%

Table 6: Cpu time and bounds improvements for several trees

5 Conclusions

In this paper we provide an algorithm to compute lower bounds for the Kirchhoff index taking into account additional information on the localization of the eigenvalues of the transition matrix of the graph. To this aim upper and lower bounds of the eigenvalues are derived by means of the solution of a class of suitable nonlinear optimization problems based on majorization techniques. Some numerical examples show how sharper bounds can be obtained with respect to those existing in literature. In particular, the comparison has been done randomly generating both bipartite and non-bipartite graphs with a

different number of vertices. We point out that best results of non-bipartite graphs have been obtained considering the additional information given by the upper bound on the last eigenvalue of the transition matrix. For what concerns bipartite graphs, only the additional information on the lower bound of the second eigenvalue of the transition matrix provides significant results leading to a tighter bound.

Appendix A. *MATLAB*® code

The following function generates randomly a graph mainly based on Matgraph package[3]. The type of the graph is requested as an input. Either $G(n,p)$ or $G(n,m)$ ER variants are used to generate simple non-bipartite graphs. Path and Tree, through the Prüfer[4] code, can be derived too. Finally, the code assures that the graph is connected. The function gives back as output the adjacency matrix A of the generated graph. In the numerical application presented in Section 4 we consider all the graphs here presented.

```
function [A]=GenerateRandGraph(type,vert,q,m)

g = graph
done=false
while ~ done
switch type
    case 1
        %ER G(n,p)
        random(g,vert,q)
    case 2
        %ER Tree(n)
        random_tree(g,vert)
    case 3
        %Path(n)
        path(g,vert)
    case 4
        %ER G(n,m)
        A=zeros(vert,vert);
        while sum(sum(A))/2 < m
          i=randint(1,1,[1,vert]); j=randint(1,1,[1,vert]);
          if i==j | A(i,j)>0; continue; end  % avoid self-loops or double edges
          A(i,j)=A(i,j)+1; A(j,i)=A(i,j);
        end
          fast_set_matrix(g,logical(A));
end
done=isconnected(g);
end
A=matrix(g);
free(g)
```

The following function evaluates the lower bounds through formulae (5) and (7) for simple and connected graphs by using the procedure previously described for a defined number of vertices. In a similar way, the lower bound through formula (8) is obtained

[3]See Matgraph package by E. R. Scheinerman for functions' details

[4]The Prüfer sequence of a labeled tree is a unique sequence associated to the tree. The sequence for a tree on n vertices has length $n-2$ and it can be generated by a simple iterative algorithm. It is a way to map bijectively trees on n vertices into $n-2$ long sequences of integers drawn from n.

for bipartite graphs. As an input, the type of graph and the maximum value of p to be computed (i.e. p^*) is also required. Some characteristics of the graph (number of vertices, maximum degree d_1 and number of edges), the value of $K(G)$ and a comparison with the bounds (16), (17) or 18 are provided as main outputs.

```
function [out]=LBKirchhoffIndex(type,vert,pstar,q,m)
tic
A=GenerateRandGraph(type,vert,q,m);
D=diag(sum(A,2),0);
deg=max(diag(D));
P=inv(D)*A;
L=D-A;


a=vert;
n=a;
b=zeros(1,pstar);
lb_nu=zeros(vert-1,pstar-1);
ub_nu=zeros(vert-1,pstar-1);
for p=2:pstar
    for k=0:p
    b(p)=b(p)+factorial(p)/(factorial(k)*factorial(p-k))*(-1)^(p-k)*(trace(P^(p-k))-1);
    end

    if b(p)==(a^p)/((n-1)^(p-1))
        lb_nu(:,p-1)=(a/(n-1)).*ones(vert-1,1);
        ub_nu(:,p-1)=(a/(n-1)).*ones(vert-1,1);
    else
        if b(p)>0
            hstar=floor(nthroot(a^p/b(p),p-1));
            if hstar<(n-1)
                %Compute Upper Bounds of nu
                for h=1:vert-1
                    if hstar<h
                        func=@(alfa)(h-1)*(alfa^p)+(a-h*alfa+alfa)^p-b(p);
                        ub_nu(h,p-1)=fzero(func,[+10^(-20),a/h]);
                    else
                        func=@(alfa)h*(alfa^p)+(n-1-h)*((a-h*alfa)^p)/((n-1-h)^p)-b(p);
                        ub_nu(h,p-1)=fzero(func,[a/(n-1)+10^(-20),a/h]);
                    end
                    %Compute Lower Bounds of nu
                    if h==1
                        func=@(alfa)hstar*(alfa^p)+(a-hstar*alfa)^p-b(p);
                        lb_nu(h,p-1)=fzero(func,[a/(hstar+1),a/hstar]);
                    else if (h<=hstar+1)
                         func=@(alfa)(n-h)*(alfa^p)+(h-1)*((a-(n-h)*alfa)^p)/((h-1)^p)-b(p);
                         lb_nu(h,p-1)=fzero(func,[10^(-20),a/(n-1)]);
                     end
                   end
                end
            end
        end
    end
end


lambda=sort(round(eig(L).*10^10)./(10^10),'descend');
KG=sum(1./lambda(1:end-1),1).*(vert);

switch type
    case {1,4}
        LBEq5=vert/deg*(1/max(lb_nu(1,:))+(vert-2)^2/(vert-max(lb_nu(1,:))));
        LBEq7=vert/deg*(1/min(ub_nu(end,:))+(vert-2)^2/(vert-min(ub_nu(end,:))));
        sigma=sqrt(trace(P^2)/vert-(trace(P)/vert).^2);
        LBEq17=vert/deg*(1/(1+sigma/sqrt(vert-1))+(vert-2)^2/(vert-1-sigma/sqrt(vert-1)));
        LBEq16=(vert-1)^2/deg;
        out=[vert deg sum(sum(triu(A,0)),2) KG LBEq5 LBEq7  LBEq16 LBEq17];
    case {2,3}
        LBEq16=(vert-1)^2/deg;
```

```
        LBEq8=vert/deg*(1/2+1/min(ub_nu(end,:))+(vert-3)^2/(vert-2-min(ub_nu(end,:)))));
        LBEq18=vert*(2*vert-3)/(2*deg);
        out=[vert deg sum(sum(triu(A,0)),2) KG LBEq8  LBEq16 LBEq18];
end
toc

end
```

For further details about the $MATLAB^{®}$ code, please feel free to contact any of the authors.

References

[1] M. Bianchi, A. Cornaro, A. Torriero, Majorization under constraints and bounds of the second Zagreb index, *Math. Ineq. Appl.* **16** (2013) 329–347.

[2] M. Bianchi, A. Cornaro, A. Torriero, A majorization method for localizing graph topological indices, *Discr. Appl. Math.* **161** (2013) 2731–2739.

[3] M. Bianchi, A. Cornaro, J. L. Palacios, A. Torriero, Bounds for the Kirkhhoff index via majorization techniques, *J. Math. Chem.* **51** (2013) 569–587.

[4] M. Bianchi, A. Torriero, Some localization theorems using a majorization technique, *J. Inequ. Appl.* **5** (2000) 433–446.

[5] M. Bianchi, A. Torriero, *A Majorization Approach for Solving Nonlinear Optimization Problems*, in Convessità e calcolo parallelo, Libreria Universitaria Editrice, Verona, 1997.

[6] B. Bollobás, *Random Graphs*, Cambridge Univ. Press, London, 2001.

[7] H. Chen, F. Zhang, Resistance distance and the normalized Laplacian spectrum, *Discr. Appl. Math.* **155** (2007) 654–661.

[8] F. R. K. Chung, L. Lu, V. Vu, The spectra of random graphs with given expected degrees, *Proc. Nat. Acad. Sci.* **100** (2003) 6313–6318.

[9] I. Gutman, B. Mohar, The quasi–Wiener and the Kirchhoff indices coincide, *J. Chem. Inf. Comput. Sci.* **36** (1996) 982–985.

[10] P. Erdös, A. Rényi, On the evolution of random graphs, *Publ. Math. Inst. Hung. Acad. Sci.* **5** (1960) 17–61.

[11] P. Erdös, A. Rényi, On Random Graphs I, *Publ. Math.* **6** (1959) 290–297.

[12] E. Estrada, *The Structure of Complex Networks: Theory and Applications*, Oxford Univ. Press, New York, 2011.

[13] D. J. Klein, M. Randić, Resistance distance, *J. Math. Chem.* **12** (1993) 81–95.

[14] I. Lukovits, S. Nikolić, N. Trinajstić, Resistance distance in regular graphs, *Int. J. Quantum Chem.* **71** (1999) 217–225.

[15] J. L. Palacios, J. M. Renom, Broder and Karlin's formula for hitting times and the Kirchhoff index, *Int. J. Quantum Chem.* **111** (2011) 35–39.

[16] J. L. Palacios, J. M. Renom, Bounds for the Kirchhoff index of regular graphs via the spectra of their random walks, *Int. J. Quantum Chem.* **110** (2010) 1637–1641.

[17] R. J. Wilson, *Introduction to Graph Theory*, Addison–Wesley, Reading, 1996.

[18] G. P. Wolkowicz, G. P. H. Styan, Bounds for eigenvalues using trace, *Lin. Algebra Appl.* **29** (1980) 471–506.

[19] W. Xiao, I. Gutman, Resistance distance and Laplacian spectrum, *Theor. Chem. Acc.* **110** (2003) 284–289.

[20] Y. Yang, H. Zhang, D. Klein, New Nordhaus–Gaddum–type results for the Kirchhoff index, *J. Math. Chem.* **49** (2011) 1–12.

[21] H. P. Zhang, Y. J. Yang, Resistance distance and Kirchhoff index in circulant graphs, *Int. J. Quantum Chem.* **107** (2007) 330–339.

[22] B. Zhou, N. Trinajstić, A note on Kirchhoff index, *Chem. Phys. Lett.* **455** (2008) 120–123.

[23] H. Y. Zhu , D. J. Klein, I. Lukovits, Extensions of the Wiener number, *J. Chem. Inf. Comput. Sci.* **36** (1996) 420–428.